OOOO

Nordic and Baltic POSIX Locales and Character Sets

INSTA/IT document

Edited by Keld Simonsen
Danish Standards Association, 1992-09-19

In this specification, the cultural dependent items described in the POSIX standard (ISO/IEC 9945-2 2CD 1991) are described for the Nordic Countries: Denmark, Finland, Iceland, Norway, and Sweden, and also for the three Baltic countries: Estonia, Latvia and Lithauania. Also the character sets in common use in these countries are specified.

This specification consists of 7 parts, Part 1 describes the POSIX locale format and with the same wording as the POSIX standard, except that the references to specific POSIX operation, tables and the POSIX locale is not included. Part 2 specifies the Nordic locales. Part 3 describes a format to specify character sets and part 4 specifies character sets in use in the Nordic countries. Part 5 contains a description of character mnemonics and a definition of characters mnemonics with the encoding of the ISO 10646 UCS standard. Part 6 contains an enhanced general POSIX locale described with the mnemonics. Part 7 contains the references.

1. POSIX Locale

A locale is the definition of the subset of a user's environment that depends on language and cultural conventions. It is made up from one or more categories. Each category is identified by its name and controls specific aspects of the behaviour of components of the system. Category names correspond to the following environment variable names:

LC_CTYPE	Character classification and case conversion.
LC_COLLATE	Collation order.
LC_TIME	Date and time formats.
LC_NUMERIC	Numeric, non monetary formatting.
LC_MONETARY	Monetary formatting.
LC_MESSAGES	Formats of informative and diagnostic messages and interactive responses.

1.1. Locale Rationale

The description of locales is based on work performed in the Uniform Technical Committee Subcommittee on Internationalisation. Wherever appropriate, keywords were taken from the C Standard (15) or the X/Open Portability Guide (17).

POSIX compiles locale sources into object files. The "object" definitions need not be portable, as long as "source" definitions are. Strictly speaking, "source" definitions are portable only between implementations using the same character set(s). Such "source" definitions can, if they use symbolic names only, easily be ported between systems using different code sets as long as the characters in the portable character set (ISO 646) have common values between the code sets; this is frequently the case in historical implementatons. Of course, this requires that the symbolic names used for characters outside the portable character set are identical between character sets.

1.2. Locale Definition

Locales can be described with the file format presented in this subclause. For the purposes of this subclause, the file is referred to as the locale definition file.

The locale definition file shall contain one or more locale category source definitions, and shall not contain more than one definition for the same locale category. If the file contains source definitions for more than one category, implementation-defined categories, if present, shall appear after the categories defined by this clause. A category source definition shall contain either the definition of a category or a copy directive. In the event that some of the information for a locale category, as specified in this standard, is missing from the locale source definition, the behavior of that category, if it is referenced, is unspecified.

A category source definition shall consist of a category header, a category body, and a category trailer. A category header shall consist of the character string naming of the category, beginning with the characters LC_. The category trailer shall consist of the string END, followed by one or more <blank>s and the string used in the corresponding category header.

The category body shall consist of one or more lines of text. Each line shall contain an identifier, optionally followed by one or more operands. Identifiers shall be either keywords, identifying a particular locale element, or collating elements. In addition to the keywords defined in this standard, the source can contain implementation-defined keywords. Each keyword within a locale shall have a unique name (i.e., two categories cannot have a commonly-named keyword); no keyword shall start with the characters LC_. Identifiers shall be separated from the operands by one or more <blank>s.

Operands shall be characters, collating elements, or strings of characters. Strings shall be enclosed in double-quotes. Literal double-quotes within strings shall be preceded by the <escape character>, described below. When a keyword is followed by more than one operand, the operands shall be separated by semicolons; <blank>s shall be allowed before and/or after a semicolon. The first category header in the file can be preceded by a line modifying the comment character. It shall have the following format, starting in column 1:

"comment_char %c\n", <comment character>

The comment character shall default to the number-sign (#). Blank lines and lines containing the <comment char> in the first position shall be ignored. The first category header in the file can be preceded by a line modifying the escape character to be used in the file. It shall have the following format, starting in column 1:

"escape_char %c \n", <escape character>

The escape character shall default to backslash.

A line can be continued by placing an escape character as the last character on the line; this continuation character shall be discarded from the input. Although the implementation need not accept any one portion of a continued line with a length exceeding "LINE_MAX" bytes, it shall place no limits on the accumulated length of the continued line. Comment lines shall not be continued on a subsequent line using an escaped <newline>.

Individual characters, characters in strings, and collating elements shall be represented using symbolic names, as defined below. In addition, characters can be represented using the characters themselves, or as octal, hexadecimal, or decimal constants. When nonsymbolic notation is used, the resultant locale definitions need not be portable between systems. The left angle bracket (<) is a reserved symbol, denoting the start of a symbolic name; when used to represent itself it shall be preceded by the escape character. The following rules apply to character representation:

(1) A character can be represented via a symbolic name, enclosed within angle brackets (< and >). The symbolic name, including the angle brackets, shall exactly match a symbolic name defined in the charmap file specified via the localedef -f option, and shall be replaced by a character value determined from the value associated with the symbolic name in the charmap file. The use of a symbolic name not found in the charmap file shall constitute an error, unless the category is LC_CTYPE or LC_COLLATE, in which case it shall constitute a warning condition (see localedef for a description of action resulting from errors and , warnings). The specification of a symbolic name in a collating-element or collating-symbol clause that duplicates a symbolic name in the charmap file (if present) is an error. Use of the escape character or a right angle bracket within a symbolic name shall be invalid unless the character is preceded by the escape character.

    Example: <c>;<c-cedilla> "<M><a><y>"

(2) A character can be represented by the character itself, in which case the value of the character is implementation defined. Within a string, the double-quote character, the escape character, and the right angle bracket character shall be escaped (preceded by the escape character) to be interpreted as the character itself. Outside strings, the characters

    , ; < > escape_char

    shall be escaped to be interpreted as the character itself

Example: c ß "May"

(3) A character can be represented as an octal constant. An octal constant shall be specified as the escape character followed by two or more octal digits. Each constant shall represent a byte value. Multibyte characters can be represented by concatenated constants.

Example: \143; \347; \143\150 "\115\141\171"

(4) A character can be represented as a hexadecimal constant. A hexadecimal constant shall be specified as the escape character followed by an x followed by two or more hexadecimal digits. Each constant shall represent a byte value. Multibyte characters can be represented by concatenated constants.

Example: \x63;\xe7;\x63\x68 "\x4d\x61\x79"

(5) A character can be represented as a decimal constant. A decimal constant shall be specified as the escape character followed by a d followed by two or more decimal digits. Each constant shall represent a byte value. Multibyte values can be represented by concatenated constants.

Example: \d99; \d231; \d99\d104 "\d77\d97\d121"

Implementations may accept single-digit octal, decimal, or hexadecimal constants following the escape character. Only characters existing in the character set for which the locale definition is created shall be specified, whether using symbolic names, the characters themselves, or octal, decimal, or hexadecimal constants. If a charmap file is present, only characters defined in the charmap can be specified using octal, decimal, or hexadecimal constants. Symbolic names not present in the charmap file can be specified and shall be ignored, as specified under item (1) above.

1.2.1. Locale Definition Rationale

The decision to separate the file format from the localedef utility description was only partialy editorial. Implementations may provide other interfaces than localedef. Reqirements on "the utility", mostly concerning error messages, are described in this way because they are meant to affect the other interfaces implementations may provide as well as localedef.

To avoid confusion between an octal constant and a backreference, the octal, hexadecimal, and decimal constants must contain at least two digits. As single-digit constants are relatively rare, this should not impose any significant hardship. Each of the constants includes "two or more" digits to account for systems in which the byte size is larger than eight bits. For example, a Unicode system that has defined 16-bit bytes may require six octal, four hexadecimal, and five decimal digits.

This standard is intended as an international (ISO/IEC) standard a well as an IEEE standard, and must therefore follow the ISO/IEC guidelines. One such rule is that characters outside the invariant part of ISO/IEC 646 should not be used in portable specifications. The backslash character is not in the invariant part; the number-sign is, but with multiple representations: as a number-sign and as a pound sign. As far as general usage of these synbols, they are covered by the "grandfather clause", but for newly defined interfaces, ISO has requested that POSIX provides alternate representations. Consequently, while the default escape character remains the backslash, and the default comment character is the number-sign, implementations are required to recognize alternative representations, identified in the applicable source file via the escape_char and comment_char keywords.

1.3. LC_CTYPE

The LC_CTYPE category shall define character classification, case conversion, and other character attributes. In addition, a series of characters can be represented by three adjacent periods representing an ellipsis symbol ("..."). The ellipsis specification shall be interpreted as meaning that all values between the values preceding and following it represent valid characters. The ellipsis specification only shall be valid within a single encoded character set. An ellipsis shall be interpreted as including in the list all characters with an encoded value higher than the encoded value of the character preceding the ellipsis and lower than the encoded value of the character following the ellipsis.

Example: \x30:...;\x39; includes in the character class all characters with encoded values between the endpoints.

The following keywords shall be recognized. In the descriptions, the term "automatically included" means that it shall not be an error to either include the referenced characters or to omit them; the implementation shall provide them if missing and accept them silently if present.

copy Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.

upper Define characters to be classified as uppercase letters. No charr specified for the keywords cntrl, digit, punct, or space shall be specified. If this keyword is not specified, the uppercase letters A through Z, shall automatically belong to this class, with implementation-defined character values.

lower Define characters to be classified as lowercase letters. No character specified for the keywords cntrl, digit, punct, or space shall be specified.

If this keyword is not specified, the lowercase letters a through z, shall automatically belong to this class, with implementation-defined character values.

alpha
: Define characters to be classified as letters. No character specified for the keywords cntrl, digit, punct, or space shall be specified. In addition, characters classified as either upper or lower shall automatically belong to this class.

digit
: Define the characters to be classified as numeric digits. Only the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 shall be specified, and in ascending sequence by numerical value. If this keyword is not specified, the digits 0 through 9, shall automatically belong to this class, with implementation-defined character values.

space
: Define characters to be classified as white-space characters. No character specified for the keywords upper, lower, alpha, dlgit, graph, or xdigit shall be specified. If this keyword is not specified, the characters <space>, <form-feed>, <newline>, <carriage-return>, <tab>, and <vertical-tab>, shall automatically belong to this class, with implementation-defined character values. Any characters included in the class blank shall be automatically included.

cntrl
: Define characters to be classified as control characters. No character specified for the keywords upper, lower, alpha, dlgit, punct, graph, print, or xdigit shall be specified.

punct
: Define characters to be classified as punctuation characters. No character specified for the keywords upper, lower, alpha, digit, cntrl, xdigit, or as the <space> character shall be specified.

graph
: Define characters to be classified as printable characters, not including the <space> character. If this keyword is not specified, characters specified for the keywords upper, lower, alpha, digit, xdigit, and punct shall belong to this character class. No character specified for the keyword cntrl shall be specified.

print
: Define characters to be classified as printable characters, including the <space> character. If this keyword is not provided, characters specified for the keywords upper, lower, alpha, digit, xdigit, punct, and the <space> character shall belong to this character class. No character specified for the keyword cntrl shall be specified.

xdigit
: Define the characters to be classified as hexadecimal digits. Only the characters defined for the class digit shall be specified, in ascending sequence by numerical value, followed by one or more sets of six characters representing the hexadecimal digits 10 through 15, with each set in ascending order (for example A, B, C, D, E, F, a, b, c, d, e, f). If this keyword is not specified, the digits 0 through 9, the uppercase letters A through F, and the lowercase letters a through f, shall automatically belong to this class, with implementation-defined character values.

blank
: Define characters to be classified as <blank> characters. If this keyword is unspecified, the characters <space> and <tab> shall belong to this character class.

toupper
: Define the mapping of lowercase letters to uppercase letters. The operand shall consist of character pairs, separated by semicolons. The characters in each character pair shall be separated by a comma and the pair enclosed by parentheses. The first character in each pair shall be the lowercase letter, the second the corresponding uppercase letter. Only characters specified for the keywords lower and upper shall be specified. If this keyword is not specified, the lowercase letters a through z, and tneir corresponding uppercase letters A through Z, shall automatically be included, with implementation-defined character values.

tolower
: Define the mapping of uppercase letters to lowercase letters. The operand shall consist of character pairs, separated by semicolons. The characters in each character pair are separated by a comma and the pair enclosed by parentheses. The first character in each pair shall be the uppercase letter, the second the corresponding lowercase letter. Only characters specified for the keywords lower and upper shall be specified. The tolower keyword is optional. If specified, the uppercase letters A through Z, and their corresponding lowercase letter, shall be specified. If this keyword is not specified, the mapping shall be the reverse mapping of the one specified for toupper.

Table 2-6 shows the allowed character class combinations.

Table 2-6 Valid Character Class Combinations

In Class	Can also belong to									
	upper	lower	alpha	digit	space	cntrl	punct	graph	print	xdigit
upper	-	-	M	X	X	X	X	D	-	X
lower	-	-	M	X	X	X	X	D	D	-
alpha	-	-	-	X	X	X	X	D	D	-
digit	X	X	X	-	X	X	X	D	D	-
space	X	X	X	X	-	-	*	*	*	X
cntrl	X	X	X	X	-	-	X	X	X	X
punct	X	X	X	X	-	X	-	D	D	X
graph	-	-	-	-	-	X	-	-	-	-
print	-	-	-	-	-	X	-	-	-	-
xdigit	-	-	-	-	X	X	X	D	D	-
blank	X	X	X	X	M	-	*	*	*	X

NOTES:
(1) Explanation of codes:
M Always
D Default; belongs to class if not specified
- Permitted
X Mutually exclusive
* See note (2)

(2) The <space> character, which is part of the space and blank class, cannot belong to punct or graph, but automatically shall belong to the print class. Other space or blank characters can be classified as punct, graph, and/or print.

1.3.1. LC_CTYPE Rationale

The LC_CTYPE category primarily is used to define the encoding-independent aspects of a character set, such as character classification. In addition, certain encoding-dependent characteristics are also defined for an application via the LC_CTYPE category. POSIX.2 does not mandate that the encoding used in the locale is the same as the one used by the application, because an implementation may decide that it is advantageous to define locales in a system-wide encoding rather than having multiple, logically identical locales in different encodings, and to convert from the application encoding to the system-wide encoding on usage. Other implementations could require encoding-dependent locales. In either case, the LC_CTYPE attributes that are directly dependent on the encoding, such as mb_cur_max and the display width of characters, are not user-specifiable in a locale source, and are consequently not defined as keywords.

As the LC_CTYPE character classes are based on the C Standard (2) character-class definition, the category does not support multicharacter elements. For instance, the German character <sharp-s> is traditionally classified as a lowercase letter. There is no corresponding uppercase letter; in proper capitalization of German text the <sharp-s> will be replaced by SS; i.e., by two characters. This kind of conversion is outside the scope of the toupper and tolower keywords. Where POSIX.2 specifies that only certain characters can be specified, as for the keywords dlgit and xdigit, the specified characters must be from the portable character set, as shown. As an example, only the Arabic digits 0 through 9 are acceptable as digits.

The character classes digit, xdigit, lower, upper, and space have a set of automatically included characters. These only need to be specified if the character values (i.e. encoding) differs from the implementation default values. The definition of character class digit requires that only ten characters--the ones defining digits--can be specified; alternate digits (e.g., Hindi or Kanji) cannot be specified here. However, the encoding may vary if an implementation supports more than one encoding. The definition of character class xdigit requires that the characters included in character class digit are included here also, and allows for different symbols for the hexadecimal digits 10 through 15.

1.4. LC_COLLATE

A collation sequence definition shall define the relative order between collating elements (characters and multicharacter collating elements) in the locale. This order is expressed in terms of collation values; i.e., by assigning each element one or more collation values (also known as collation weights). This does not imply that implementations shall assign such values, but that ordering of strings using the resultant collation definition in the locale shall behave as if such assignment is done and used in the collation process. The collation sequence definition shall be used by regular expressions, pattern matching, and sorting. The following capabilities are provided:

(1) Multicharacter collating elements. Specification of multicharacter collating elements (i.e., sequences of two or more characters to be collated as an entity).

(2) User-defined ordering of collating elements. Each collating element shall be assigned a collation value defining its order in the character (or basic) collation sequence. This ordering is used by regular expressions and pattern matching and, unless collation weights are explicitly specified, also as the collation weight to be used in sorting.

(3) Multiple weights and equivalence classes. Collating elements can be assigned one or more (up to the limit (COLL_WEIGHTS_MAX)) collating weights for use in sorting. The first weight is hereafter referred to as the primary weight.

(4) One-to Many mapping. A single character is mapped into a string of collating elements.

(5) Many-to-Many substitution. A string of one or more characters is substituted by another string (or an empty string, i.e., the character or characters shall be ignored for collation purposes).

(6) Equivalence class definition. Two or more collating elements have the same collation value (primary weight).

(7) Ordering by weights. When two strings are compared to determine their relative order, the two strings are first broken up into a series of collating elements, and each successive pair of elements are compared according to the relative primary weights for the elements. If equal, and more than one weight has been assigned, then the pairs of collating elements are recompared according to the relative subsequent weights, until either a pair of collating elements compare unequal or the weights are exhausted.

The following keywords shall be recognized in a collation sequence definition. They are described in detail in the following subclauses.

copy — Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.

collating-element — Define a collating-element symbol representing a multicharacter collating element. This keyword is optional.

collating-symbol — Define a collating symbol for use in collation order statements. This keyword is optional.

order_start — Define collation rules. This statement is followed by one or more collation order statements, assigsing character collation values and collation weights to collating elements.

order_end — Specify the end of the collation-order statements.

1.4.1. collating-element Keyword

In addition to the collating elements in the character set, the collating-element keyword shall be used to define multicharacter collating elements. The syntax is

"collating-element %s from %s\n",<collating-symbol>,<string>

The <collating-symbol> operand shall be a symbolic name, enclosed between angle brackets (< and >), and shall not duplicate any symbolic name in the current charmap file (if any), or any other symbolic name defined in this collation definition. The string operand shall be a string of two or more characters that shall collate as an entity. A <collating-element> defined via this keyword is only recognized with the LC_COLLATE category.

Example:
collating-element <ch> from <c><h>
collating-element <e-acute> from <acute><e>
collating-element <ll> from ll

1.4.2. collating-symbol Keyword

This keyword shall be used to define symbols for use in collation sequence statements; i.e., between the order_start and the order_end keywords. The syntax is

"collating-symbol %s\n", <collating-symbol>

The <collating-symbol> shall be a symbolic name, enclosed between angle brackets (< and >), and shall not duplicate any symbolic name in the current charmap file (if any), or any other symbolic name defined in this collation definition. A <collating-symbol> defined via this keyword is only recognized with the LC_COLLATE category.

Example:
collating-symbol <UPPER_CASE>
collating-symbol <HIGH>

1.4.3. order_start Keyword

The order_start keyword shall precede collation order entries and also defines the number of weights for this collation sequence definition and other collation rules.

The syntax of the order_start keyword is:

"order_start %s;%s;...;%s\n", <sort-rules>, <sort-rules> ...

The operands to the order_start keyword are optional. If present, the operands define rules to be applied when strings are compared. The number of operands define how many weights each element is assigned; if no operands are present, one forward operand is assumed. If present, the first operand defines rules to be applied when comparing strings using the first (primary) weight; the second when comparing strings using the second weight, and so on. Operands shall be separated by semicolons (;). Each operand shall consist of one or more collation directives, separated by commas (,). If the number or operands exceeds the (COLL_WEIGHTS_MAX) limit, the utility shall issue a warning message. The following directives shall be supported:

forward Specifies that comparison operations for the weight level shall proceed from start of string towards the end of string.

backward Specifies that comparison operations for the weight level shall proceed from end of string towards the beginning of string.

position Specifies that comparison operations for the weight level will consider the relative position of non-IGNOREd elements in the strings. The string containing a non-IGNOREd element after the fewest IGNOREd collating elements from the start of the compare shall collate first. If both strings contain a non-IGNOREd character in the same relative position, the collating values assigned to the elements shall determine the ordering. In case of equality, subsequent non-IGNOREd characters shall be considered in the same manner.

The directives forward and backward are mutually exclusive.

Example:
order_start forward;backward

If no operands are specified, a single forward operand shall be assumed.

1.4.4. Collation Order

The order_start keyword shall be followed by collating element entries. The syntax for the collating element entries is

"%s %s;%s;...;%s\n",<collating-element>,<weight>,<weight>,...

Each collating-element shall consist of either a character (in any of the forms defined in 2.5.2), a <collating-element>, a <collating-symbol>, an ellipsis, or the special symbol UNDEFINED. The order in which collating elements are specified determines the character collation sequence, such that each collating element shall compare less than the elements following it. The NUL character shall compare lower than any other character.

A <collating-element> shall be used to specify multicharacter collating elements, and indicates that the character sequence specified via the <collating-element> is to be collated as a unit and in the relative order specified by its place.

A <collating-symbol> shall be used to define a position in the relative order for use in weights.

The ellipsis symbol ("...") specifies that a sequence of characters shall collate according to their encoded character values. It shall be interpreted as indicating that all characters with a coded character set value higher than the value of the character in the preceding line, and lower than the coded character set value for the character in the following line, in the current coded character set, shall be placed in the character collation order between the previous and the following character in ascending order according to their coded character set values. An initial ellipsis shall be interpreted as if the preceding line specified the NUL character, and a trailing ellipsis as if the following line specified the highest coded character set value in the current coded character set. An ellipsis shall be treated as invalid if the preceding or following lines do not specify characters in the current coded character set. The use of the ellipsis symbol ties the definition to a specific coded character set and may preclude the definition from being portable between implementations.

The symbol UNDEFINED shall be interpreted as including all coded character set values not specified explicitly or via the ellipsis symbol. Such characters shall be inserted in the character collation order at the point indicated by the symbol, and in ascending order according to their coded character set values. If no UNDEFINED symbol is specified, and the current coded character set contains characters not specified in this clause, the utility shall issue a warning message and place such characters at the end of the character collation order.

The optional operands for each collation-element shall be used to define the primary, secondary, or subsequent weights for the collating element. The first operand specifies the relative primary weight, the second the relative secondary weight, and so on. Two or more collation-elements can be assigned the same weight; they belong to the same equivalence class if they have the same primary weight. Collation shall behave as if, for each weight level, IGNOREd elements are removed. Then each successive pair of elements shall be compared according to the relative weights for the elements. If the two strings compare equal, the process shall be repeated for the next weight level, up to the limit "COLL_WEIGHTS_MAX" .

Weights shall be expressed as characters (in any of the forms specified here), <collating-symbol>s, <collating-element>s, an ellipsis, or the special symbol IGNORE. A single character, a <collating-symbol>, or a <collating-element> shall represent the relative order in the character collating sequence of the character or symbol, rather than the character or characters themselves.

One-to-many mapping is indicated by specifying two or more concatenated characters or symbolic names. Thus, if the character <eszet> is given the string <s><s> as a weight, comparisons shall be performed as if all occurrences of the character <eszet> are replaced by <s><s>. If it is desirable to define <eszet> and <s><s> as an equivalence class, then a collating-element must be defined for the string "ss", as in the example below.

All characters specified via an ellipsis shall by default be assigned unique weights, equal to the relative order of characters. Characters specified via an explicit or implicit UNDEFINED special symbol shall by default be assigned the same primary weight (i.e., belong to the same equivalence class). An ellipsis symbol as a weight shall be interpreted to mean that each character in the sequence shall have unique weights, equal to the relative order of their character in the character collation sequence. Secondary and subsequent weights have unique values. The use of the ellipsis as a weight shall be treated as an error if the collating element is neither an ellipsis nor the special symbol UNDEFINED.

The special keyword IGNORE as a weight shall indicate that when strings are compared using the weights at the level where IGNORE is specified, the collating element shall be ignored; i.e., as if the string did not contain the collating element. In regular expressions and pattern matching, all characters that are IGNOREd in their primary weight form an equivalence class.

An empty operand shall be interpreted as the collating-element itself.

For example, the order statement

<a> <a>;<a>

is equal to

<a>

An ellipsis can be used as an operand if the collating-element was an ellipsis, and shall be interpreted as the value of each character defined by the ellipsis.

The collation order as defined in this clause defines the interpretation of bracket expressions in regular expressions (see POSIX.2 2.8.3.2 - it is not included in this Nordic & Baltic locales draf.).

Example:

```
order_start     forward;backward
UNDEFINED       IGNORE;IGNORE
<LOW>
<space>         <LOW>; <space>
...             <LOW>;
<a>             <a>;<a>
<a-acute>       <a>;<a-acute>
<a-grave>       <a>;<a-grave>
<A>             <a>;<A>
<A-acute>       <a>;<A-acute>
<A-grave>       <a>;<A-grave>
<Ch>            <Ch>;<Ch>
<Ch>            <Ch>;<Ch>
<s>             <s>;<s>
<eszet>         <s><s>;<eszet><eszet>
...             <HIGH>;
                <HIGH>
order_end
```

This example is interpreted as follows:

(1) The UNDEFINED means that all characters not specified in this definition (explicitly or via the ellipsis) shall be ignored for collation purposes; for regular expression purposes they are ordered first.

(2) All characters between <space> and <a> shall have the same primary equivalence class and individual secondary weights based on their ordinal encoded values.

(3) All characters based on the upper or lowercase character "a" belong to the same primary equivalence class.

(4) The multicharacter collating element <c><h> is represented by the collating symbol <ch> and belongs to the same primary equivalence class as the multicharacter collating element <C><h>.

(5) Note that it is not possible to use the collating element <ss> as a weight and expect it to be espanded to the string "ss". When used as a weight, any

collating-element represents the relative order assigned to it in the character collation sequence, not the string from which it was derived (compare with <ch>).

1.4.5. order_end Keyword

The collating order entries shall be terminated with an order_end keyword.

1.4.6. LC_COLLATE Rationale.

The LC_COLLATE category governs the collation order in the locale, and thus the processing of the C Standard (15) strxfrm() and strcoll() functions, as well as a number of POSIX.2 utilities.

The rules governing collation depends to some extent on the use. At least five different levels of increasingly complex collation rules can be distinguished:

(1) Byte/machine code order. This is the historical collation order in the UNIX system and many proprietary operating systems. Collation is here done character by character, without any regard to context. The primary virtue is that it usually is quite fast, and also completely deterministic; it works well when the native machine collation sequence matches the user expectations.

(2) Character order. On this level, collation is also done character by charac ter, without regard to context. The order between characters is, however, not determined by the code values, but on the user's expectations of the correct order between characters. In addition, such a (simple) collation order can specify that certain characters collate equal (e.g., upper and lowercase letters).

(3) String ordering. On this level, entire strings are compared based on relatively straightforward rules. At this level, several "passes" may be required to determine the order between two strings. Characters may be ignored in some passes, but not in others; the strings may be compared in different directions; and simple string substitutions may be made before strings are compared. This level is best described as "dictionary" ordering; it is based on the spelling, not the pronunciation, or meaning, of the words.

(4) Text search ordering. This is a further refinement of the previous level, best described as "telephone book ordering"; some common homonyms (words spelled differently but with same pronunciation) are collated together; numbers are collated as if spelled with words, and so on.

(5) Semantic level ordering. Words and strings are collated based on their meaning; entire words (such as "the") are eliminated, the ordering is not deterministic. This may requires special software, and is higly dependent on the intended use.

While the historical collation order formally is at level 1, for the English language it comsponds roughly to elements at level 2. The user espects to see the output from the "ls" utility sorted very much as as it would be in a dictionary. While telephone book ordering would be an optimal goal for standard collation, this was ruled out as the order would be language dependent. Furthermore, a requirement was that the order must be determined solely from the text string and the collation rules; no external information (e.g., "pronunciation dictionaries") could be required.

As a result, the goal for the collation support is at level 3. This also matches the requirements for the proposed Canadian collation order, as well as other, known collation requirements for alphabetic scripts. It specifically rules out collation based on pronunciation rules, or based on semantic analysis of the text. The syntax for the LC_COLLATE category source is the result of a cooperative effort between representatives for many countries and organizations working with international issues, such as UniForum, X/Open, and ISO, and it meets the requirements for level 3, and has been verified to produce the correct result with examples based on French, Canadian, and Danish collation order, as well as meeting the requirements in the X/Open Portability Guide, Issue 3. (17). Because it supports multicharacter collating elements, it is also capable of supporting collation in code sets where a character is expressed using nonspacing characters followed by the base character (such as ISO 6937 (16).

The directives that can be specified in an operand to the order_start keyword are based on the requirements specified in several proposed standards and in customary use. The following is a rephrasing of rules defined for "lexical ordering in English and French" by the Canadian Standards Association (text is brackets is rephrased):

(1) Once special characters ([punctuation]) have been removed from original strings, the ordering is determined by scanning forward (left to right) [disregarding case and diacriticals].

(2) In case of equivalence, special characters are once again removed from original strings and the ordering is determined scanning backward (starting from the rightmost character of the string and back), character by character, [disregarding case but considering diacriticals].

(3) In case of repeated equivalence, special characters are removed again from original strings and the ordering is determined scanning fonvard, character by character, [considering both case and diacriticals].

(4) If there is still an ordering equivalence after rules (1) through (3) have been applied, then only special characters and the position they occupy in the string are considered to determine ordering. The string that has a special character in the lowest position comes first. If two strings have a special character in the same position, the character [with the lowest collation value] comes first. In case of equality, the other special characters are considered until there is a difference or all special characters have been exhausted.

It is estimated that the standard covers the requirements for all European languages, and no particular problems are anticipated with Slavic or Middle East character sets.

The Far East (particularly Japanese/Chinese) collations are often based on contextual information and pronunciation rules (the same ideogram can have different meanings and different pronunciations). Such collation, in general, falls outside the desired goal of the standard. There are, however, several other collation rules (stroke/radical, or "most common pronunciation") which can be supported with the mechanism described here. Previous drafts contained a substitute statement, which performed a regular expression style replacement before string compares. It has been withdrawn based on balloter objections that it was not required for the types of ordering POSIX is aimed at.

The character (and collating element) order is defined by the order in which characters and elements are specified between the order_start and order_end keywords. This character order is used in range expressions in regular expressions. Weights assigned to the characters and elements defines the collation sequence; in the absence of weights, the character order is also the collation sequence.

The position keyword was introduced to provide the capability to consider, in a compare, the relative position of non-IGNOREd characters. As an example, consider the two strings "o-ring" and "or-ing". Assuming the hyphen is IGNOREd on the first pass, the two strings will compare equal, and the position of the hyphen is immaterial. On second pass, all characters except the hyphen are IGNOREd, and in the normal case the two strings would again compare equal. By taking position into account, the first collates before the second.

1.5. LC_MONETARY

The LC_MONETARY category shall define the rules and symbols that shall be used to format monetary numeric information. The operands are strings. For some keywords, the strings can contain only integers. Keywords that are not provided, string values set to the empty string "", or integer keywords set to -1, shall be used to indicate that the value is unspecified. The following keywords shall be recognized:

copy — Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.

int_curr_symbol — The international currency symbol. The operand shall be a four character string, with the first three characters containing the alphabetic international currency symbol in accordance with those specified in ISO 4217 (18) (Codes for the representation of currencies and funds). The fourth character shall be the character used to separate the international currency symbol from the monetary quantity.

currency_symbol — The string that shall be used as the local currency symbol.

mon_decimal_point — The operand is a string containing the symbol that shall be used as the decimal delimiter in monetary formatted quantities. In contexts where other standards limit the mon_decimal_point to a single byte, the result of specifying a multibyte operand is unspecified.

mon_thousands_sep — The operand is a string containing the symbol that shall be used as a separator for groups of digits to the left of the decimal delimiter in formatted monetary quantities. In contexts where other standards limit the mon_thousands_sep to a single byte, the result of specifying a multibyte operand is unspecified.

mon_grouping — Define the size of each group of digits in formatted monetary quantities. The operand is a sequence of integers separated by semicolons. Each integer specifies the number of digits in each group, with the initial integer defining the size of the group immediately preceding the decimal delimiter, and the following integers defining the preceding groups. If the last integer is not -1, then the size of the previous group (if any) shall be repeatedly used for the remainder of the digits. If the last integer is -1, then no further grouping shall be performed.

positive_sign — A string that shall be used to indicate a nonnegative-valued formatted monetary quantity.

negative_sign — A string that shall be used to indicate a negative-valued formatted monetary quantity.

int_frac_digits — An integer representing the number of fractional digits (those to the rIght of the decimal delimiter) to be written in a formatted monetary quantity using int_curr_symbol.

frac_digits — An integer representing the number of fractional digits (those to the right of the decimal delimiter) to be written in a formatted monetary quantity using currency_symbol.

p_cs_precedes — An integer set to 1 if the currency_symbol or int_curr_symbol precedes the value for a nonnegative formatted monetary quantity, and set to 0 if the symbol succeeds the value.

p_sep_by_space — An integer set to 0 if no space separates the currency_symbol or int_curr_symbol from the value for a nonnegative formatted monetary quantity, set to 1 if a space separates the symbol from the value, and set to 2 if a space separates the symbol and the sign string, if adjacent.

n_cs_precedes — An integer set to 1 if the currency_symbol or int_curr_symbol precedes the value for a negative formatted monetary quantity, and set to 0 if the symbol succeeds the value.

n_sep_by_space — An integer set to 0 if no space separates the currency_symbol or int_curr_symbol from the value for a negative formatted monetary quantity, set to 1 if a space separates the symbol from the value, and set to 2 if a space separates the symbol and the sign string, if adjacent.

p_sign_posn — An integer set to a value indicating the positioning of the positive_sign for a nonnegative formatted monetary quantity. The following integer values shall be recognized:

0 Parentheses enclose the quantity and the currency_symbol or int_curr_symbol.

1 The sign string precedes the quantity and the currency_symbol or int_curr_symbol.

2 The sign string succeeds the quantity and the currency_symbol or int_curr_symbol.

3 The sign string immediately precedes the currency_symbol or int_curr_symbol.

4 The sign string immediately succeeds the currency_symbol or int_curr_symbol.

n_sign_posn — An integer set to a value indicating the positioning of the negative_sign for a negative formatted monetary quantity. The following integer values shall be recognized:

0 Parentheses enclose the quantity and the currency_symbol or int_curr_symbol.

1 The sign string precedes the quantity and the currency_symbol or int_curr_symbol.

2 The sign string succeeds the quantity and the currency_symbol or int_curr_symbol.

3 The sign string immediately precedes the currency_symbol or int_curr_symbol.

4 The sign string immediately succeeds the currency_symbol or int_curr_symbol.

1.5.1. LC_MONETARY Rationale.

The currency symbol does not appear in LC_MONETARY because it is not defined in the C Standard's (15) C locale.

The C Standard (15) limits the size of decimal points and thousands delimiters to single-byte values. In locales based on multibyte coded character sets this cannot be enforced, obviously; this standard does not prohibit such characters, but makes the behavior unspecified [in the text "In contexts where other standards . . . "].

The grouping specification is based on, but not identical to, the C Standard (15). The "-1" signals that no further grouping shall be performed, the equivalent of (CHAR_MAX) in the C Standard (15)).

The locale definition is an extension of the C Standard (15) localeconv() specification. In particular, rules on how currency_symbol is treated are extended to also cover int_curr_symbol, and p_set_by_space and n_sep_by_space have been augmented with the value 2, which places a space between the sign and the symbol (if they are adjacent; otherwise it should be treated as a 0). The following table shows the result of various combinations:

	p_sep_by_space 2	1	0
p_sign_posn			
p_cs_precedes=1			
0	($1.25)	($ 1.25)	($1.25)
1	+ $1.25	+$ 1.25	+$1.25
2	$1.25 +	$ 1.25+	$1.25+
3	+ $1.25	+$ 1.25	+$1.25
4	$ +1.25	$+ 1.25	$+1.25
p_cs_precedes=0			
0	(1.25 $)	(1.25 $)	(1.25$)
1	+1.25 $	+1.25 $	+1.25$
2	1.25$ +	1.25 $+	1.25$+
3	1.25+ $	1.25 +$	1.25+$
4	1.25$ +	1.25 $+	1.25$+

The following is an example of the interpretation of the mon_grouping keyword. Assuming that the value to be formatted is 123456789 and the mon_thousands_sep is "’", then the following table shows the result. The third column shows the equivalent C Standard (15) string that would be used to accommodate this grouping. It is the responsibility of the utility to perform mappings of the formats in this clause to those used by language bindings such as the C Standard (15).

Mon_grouping	Formatted Value	C String
3;-1	123456’789	"\3\177"
3	123’456’789	"\3"
3;2;-1	1234’56’789	"\3\2\177"
3;2	12’34’56’789	"\3\2"
-1	123456789	"177"

In these examples, the octal value of (CHAR_MAX) is 177.

1.6. LC_NUMERIC

The LC_NUMERIC category shall define the rules and symbols that shall be used to format nonmonetary numeric information. The operands are strings. For some keywords, the strings only can contain integers. Keywords that are not provided, string values set to the empty string (""), or integer keywords set to -1, shall be used to indicate that the value is unspecified. The following keywords shall be recognized:

copy
: Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.

decimal_point
: The operand is a string containing the symbol that shall be used as the decimal delimiter in numeric, nonmonetary formatted quantities. This keyword cannot be omitted and cannot be set to the empty string. In contexts where other standards limit the decimal point to a single byte, the result of specifying a multibyte operand is unspecified.

thousands_sep
: The operand is a string containing the symbol that shall be used as a separator for groups of digits to the left of the decimal delimiter in numeric, non-monetary formatted monetary quantities. In contexts where other standards limit the thousands_sep to a single byte, the result of specifying a multi-byte operand is unspecified.

grouping
: Define the size of each group of digits in formatted non-monetary quantities. The operand is a sequence of integers separated by semicolons. Each integer specifies the number of digits in each group, with the initial integer defining the size of the group immediately preceding the decimal delimiter, and the following integers defining the preceding groups. If the last integer is not -1, then the size of the previous group (if any) shall be repeatedly used for the remainder of the digits. If the last integer is -1, then no further grouping shall be performed.

1.6.1. LC_NUMERIC Rationale.

See the rationale for LC_MONETARY (2.5.2.3.1) for a description of the behavior of grouping.

1.7. LC_TIME

The LC_TIME category shall define the interpretation of the field descriptors supported by the date utility (see 4.15).

The following mandatory keywords shall be recognized:

copy
: Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.

abday
: Define the abbreviated weekday names, corresponding to the %a field descriptor. The operand shall consist of seven semicolon-separated strings. The first

string shall be the abbreviated name of the first day of the week (Sunday), the second the abbreviated name of the second day, and so on.

day Define the full weekday names, corresponding to the %A field descriptor. The operand shall consist of seven semicolon-separated strings. The first string shall be the full name of the first day of the week (Sunday), the second the full name of the second day, and so on.

abmon Define the abbreviated month names, corresponding to the %b field descriptor. The operand shall consist of twelve semicolon-separated strings. The first string shall be the abbreviated name of the first month of the year (January), the second the abbreviated name of the second month, and so on.

mon Define the full month names, corresponding to the %B field descriptor. The operand shall consist of twelve semicolon-separated strings. The first string shall be the full name of the first month of the year (January), the second the full name of the second month, and so on.

d_t_fmt Define the appropriate date and time representation, corresponding to the %c field descriptor. The operand shall consist of a string, and can contain any combination of characters and field descriptors. In addition, the string can contain escape sequences defined in Table 2-15.

d_fmt Define the appropriate date representation, corresponding to the %X field descriptor. The operand shall consist of a string, and can contain any combination of characters and field descriptors. In addition, the string can contain escape sequences defined in Table 2-15.

t_fmt Define the appropriate time representation, corresponding to the %X field descriptor. The operand shall consist of a string, and can contain any combination of characters and field descriptors. In addition, the string can contain escape sequences defined in Table 2-15.

am_pm Define the appropriate representation of the ante meridiem and post meridiem strings, corresponding to the %p field descriptor. The operand shall consist of two strings, separated by a semicolon. The first string shall represent the ante rneridiem designation, the last string the post meridiem designation.

t_fmt_ampm Define the appropriate time representation in the 12-hour clock format with am_pm, corresponding to the %r field descriptor. The operand shall consist of a string and can contain any combination of characters and field descriptors. If the string is empty, the 12-hour format is not supported in the locale.

It is implementation defined whether the following optional keywords shall be recognized. If they are not supported, but present in a localedef source, they shall be ignored.

era Shall be used to define alternate Eras, corresponding to the %E field descriptor modifier. The format of the operand is unspecified, but shall support the definition of the %EC and %Ey field descriptors, and may also define the era_year format (%EY).

era_year Shall be used to define the format of the year in alternate Era format, corresponding to the %EY field descriptor.

era_d_fmt Shall be used to define the format of the date in alternate Era notation, corresponding to the %Ex field descriptor.

alt_digits Shall be used to define alternate symbols for digits, corresponding to the %O field descriptor modifier. The operand shall consist of semicolon-separated strings. The first string shall be the alternate symbol corresponding with zero, the second string the symbol corresponding with one, and so on. Up to 100 alternate symbol strings can be specified. The %O modifier indicates that the string corresponding to the value specified via the field descriptor shall be used instead of the value.

1.7.1. LC_TIME Rationale.

Although certain of the field descriptors in the POSIX Locale (such as the name of the month) are shown with initial capital letters, this need not be the case in other locales. Programs using these fields may need to adjust the capitalization if the output is going to be used at the beginning of a sentence.

The LC_TIME descriptions of abday, day, and abmon imply a Gregorian style calendar (7-day weeks, 12-month years, leap years, etc.). Formatting time strings for other types of calendars is outside the scope of this standard.

As specified under the date command, the field descriptors corresponding to the optional keywords consist of a modifier followed by a traditional field descriptor (for instance %Ex). If the optional keywords are not supported by the implementation or are unspecified for the current locale, these field descriptors shall be treated as the traditional field descriptor. For instance, assume the following keywords:

```
alt_digits "0th";"1st";"2nd";"3rd";"4th";"5th";\
           "6th";"7th";"8th";"9th";"10th"
d_fmt "The %Od day of %B in %Y"
```

On 7/4/1776, the %x field descriptor would result in "The 4th day of July in 1776," while 7/14/1789 would come out as "The 14 day of July in 1789." It can be noted that the above example is for illustrative purposes only; the %o modifier is primarily intended to provide for Kanji or Hindi digits in date formats. While it is clear that an alternate year format is required, there is no consensus on the format or the requirements. As a result, while these keywords are reserved, the details are left unspecified. It is expected that National Standards Bodies will provide specifications.

1.8. LC_MESSAGES

The LC_MESSAGES category shall define the format and values for affirmative and negative responses. The operands shall be strings or extended regular expressions; see POSIX.2 2.8.4. The following keywords shall be recognized:

copy	Specify the name of an existing locale to be used as the source for the definition of this category. If this keyword is specified, no other keyword shall be specified.
yesexpr	The operand shall consist of an extended regular expression that describes the acceptable affirmative response to a question expecting an affirmative or negative response.
noexpr	The operand shall consist of an extended regular expression that describes the acceptable negative response to a question expecting an affirmative or negative response.

1.8.1. LC_MESSAGES Rationale.

The LC_MESSAGES category is described in 1.8.1 as affecting the language used by utilities for their output. The mechanism used by the implementation to accomplish this, other than the responses shown here in the locale definition file, is not specified by this version of this standard. The POSIX working group is developing an interface that would allow applications (and, presumably some of the standard utilities) to access messages from various message catalogs, tailored to a user's LC_MESSAGES value.

1.9. Date Field Descriptors

%a	Locale's abbreviated weekday name.
%A	Locale's full weekday name.
%b	Locale's abbreviated month name.
%B	Locale's full month name.
%c	Locale's appropriate date and time representation.
%C	Century (a year divided by 100 and truncated to integer) as decimal number (00-99).
%d	Day of the month as a decimal number (01-31).
%D	Date in the format mm/dd/yy.
%e	Day of the month as a decimal number (1-31 in at two-digit field with leading <space> fill).
%h	A synonym for %b.
%H	Hour (24-hour clock) as a decimal number (00-23).
%I	Hour (12-hour clock) as a decimal number (01-12).
%j	Day of the year as a decimal number (001-366).
%m	Month as a decimal number (01-12).
%M	Minute as a decimal number (00-59).
%n	A <newline> character.
%p	Locale's equivalent of either AM or PM.
%r	12-hour clock time (01-12) using the AM/PM notation; in the POSIX Locale, this shall be equivalent to "%I:%M:%S %p".
%S	Seconds as a decimal number (00-61).
%t	A <tab> character.
%T	24-hour clock time in the format HH:MM:SS.
%U	Week number of the year (Sunday as the first day of the week) as a decimal number (00-53).
%w	Weekday as a decimal number (0(Sunday)-6).
%W	Week number of the year (Monday as the first day of the week) as a decimal number (00-53).
%x	Locale's appropriate date representation.
%X	Locale's appropriate time representation.
%y	Year (offset from %C) as a decimal number (00-99).
%Y	Year with century as a decimal number.
%Z	Time-zone namen, or no characters if no time zone is determinable.

%% A <percent-sign> character.

Modified Field Descriptors

Some filed descroptors can be modified by the E and O modifier characters to indicate a different format or specification as specified in the LC_TIME locale description. If the corresponding keyword (see era, era_year, era_d_fmt, and alt_digits) is not specified or not supported for the current locale, the unmodified field descriptor value shall be used.

%Ec Locale's alternate date and time representation.

%EC The name of the base year (period) in the locale's alternate representation.

%Ex Locale's alternate date representation.

%Ey Offset from %EC (year only) in the locale's alternate representation.

%EY Full alternate year representation.

%Od Day of month using the locale's alternate numeric symbols.

%Oe Day of month using the locale's alternate numeric symbols.

%OH Hour (24-hour clock) using the locale's alternate numeric symbols.

%OI Hour (12-hour clock) using the locale's alternate numeric symbols.

%Om Month using the locale's alternate numeric symbols.

%OM Minutes using the locale's alternate numeric symbols.

%OS Seconds using the locale's alternate numeric symbols.

%OU Week number of the year (Sunday as the first day of the week) using the locale's alternate numeric symbols.

%Ow Weekday as number in the locale's alternate representation (Sunday=0).

%OW Week number of the year (Monday as the first day of the week) using the locale's alternate numeric symbols.

%Oy Year (offset from %C) in alternate representation.

2. Nordic and Baltic POSIX locales

In this chapter Nordic and Baltic POSIX locales are specified, via the "replace-after" technique also described here. The POSIX locales are specified by people from the respective official standardisation committees in these countries, but they are not normative, as work is still underway. They represent however the best effort from the standards commitees at this point in time.

The locales and other related material is available by electronic means - e-mail and FTP - from:

dkuug.dk 129.142.96.41

in the catalogue: i18n

Emial: mail archive@dkuug.dk
Subject: help

2.1. Experiences from Danish locale work

Work with the Danish locale produced a quite elaborate locale defined for a lot of character sets, including parts of ISO/IEC 10646 (currently 2nd DIS 1991) and almost all of the ISO 2375 registry (done by ECMA), and some 60 vendor specific character sets, in all about 140 character sets. The locale is available electronically together with the 140 charmaps.

Thus with just one specification of a national locale, uniform collating for many character sets is defined - the characters will always come in the same sequence regardless of which character set employed. Also there is just one definition of date format and the other cultural items to be done, and that specification is then valid for many character sets.

2.1.1. Specifying cultural data

In creating a cultural locale, many things must be considered. Some data may be easier determined than others. For instance the character classification section of the locale is normally straightforward; an "A" is considered a letter in about all languages and is mapped to an "a" when the lowercase letter should be found. Normally the LC_CTYPE definition in POSIX.2 Annex F can be used without change. The data in LC_NUMERIC, LC_MONETARY, LC_TIME and LC_MESSAGES is normally not that difficult to determine. In producing the Danish locale we had some slight problems with the date format, including time zone names, which were not well defined. We consulted as many official sources as possible, including orthography definitions and numeric rendering standards. One thing we changed late in the process was to write day names with an initial small letter - which was in accordance with the Danish orthography dictionary.

2.1.2. Collating specification considerations

But the Danish collating sequence was harder to define. There are many levels of complication for collation. For example the telephone level, with Mc the same as Mac, numbers spelled out, certain words like "the" ignored or moved to the end etc. Actually Danish has some rules like that, also in the official collating standard DS 377 from 1980. Another level is the phonetic level - soundex, which is a little less complicated. A third level is transcripted characters, as the librarians use when they see a greek alpha and order that as a normal "a".

The level that Danish Standards have decided on for its POSIX.2 locale is the systems interface level. The collating order should be usable in POSIX systems tools like ls and sort. A requirement has been that it is deterministic; if two strings are different they will also differ when compared. Another issue has been efficiency. POSIX has provisions for substituting "Mc" with "Mac", but this is considered too inefficient and avoided in the Danish example national locale.

The problem of pronounciation and translitteration has not been addressed. Instead it had been considered adequate just to look at the characters themselves - only considering characters at the systems level - and not sounds. The level provided by the Danish locale is a service for comparing strings which are intended for a replacement to the standard strcmp() etc rutines, just a little more intelligent and adhering to Danish collating rules.

We have however put as much intelligence in there as possible at this level. The two letters <a><a> are sorted as the single letter <aa> (A WITH RING), but the <aa> single letter is before <a><a> in homonyms. The 4 level scheme of the Canadian-French sorting is being used, with the four levels being letter, accent, case and special character. This was actually also specified in the DS 377. In cause of harmonization we decided to use the reverse sorting for the accents as the Canadians do; the natural choice may have been forward sorting here too, but as most of these words would be of French origin anyway, we decided to follow their rules. For <ss> we implemented what we think is the German rule, as seen in several German dictionaries. <ss> is ordered as <s><s> but before it in homonyms.

For the accents there was some indicated rules in the DS 377 and in the official Danish orthography dictionary, but it was far from complete. Then the accent sequence in several ISO standards were used, when there were no clear Danish rule. About 25 accents have been ordered.

For the non-latin scripts we decided not to transcribe. This also allows us to use the native collation order for these scripts, like alpha, beta, gamma for Greek and a be ve ghe for Cyrillic. Accented Greek and Cyrillic letters and ligatures have been put into the right places.

The sequence of the scripts was taken as in the ISO 10646 draft. That should solve the question on which scripts should come before others. Current scripts addressed are: Latin, Greek, Cyrillic, Hebrew, Arabic, Kana and special characters. Ideographic characters are in the works.

Together with the Danish collating sequence a more general collating sequence was specified. This collating sequence could be used as a reference sequence, as mentioned below, and it should produce an order which is compliant with at least English, French, German, Italian, Dutch, Portuguese, Greek, Russian, Hebrew and Arabic.

We recommend that similar decisions are taken when producing a new collating sequence.

2.2. replace-after and replace-end keywords in LC_COLLATE

For the LC_COLLATE definition, these constructs are also allowed in addition to the POSIX specifications:

```
replace-after <collating-element>
<collating-el1> ...
<collating-el2> ...
...
replace-end
```

The <collating-el1> ... are removed from the current collating sequence and inserted after <collating-element> in the collating sequence.

This construct is allowed also when a "copy" statement has been given. More than one replace-after / replace-end construct can be given.

2.2.1. replace-after rationale

Much work is done on locales, and making them quite general. WG15 RIN has on its programme of work to harmonize locales as far as it is feasible. The POSIX.2 Draft 11 introduced a copy command for all sections of the locale. This is good for many purposes and it ensures that two locales are equivalent for this category. A further step in building on previous art is proposed here.

The collating sequences vary a bit from country to country, but generally much of the collating sequence is the same. For instance the Danish sequence is quite equal to the German, English or French, but for about a dozen letters it differs. The same can be said for Swedish or Spanish: generally the latin collating sequence is the same, but a few charcters are different.

With the advent of the quite general coded character set independent locales like the example Danish in POSIX.2 draft 11 annex F, it would be convenient if the few differences could be specified just as changes to an existing one. This would also improve the overview of what the changes really are. Therefore DS has introduced the replace-after construct in the LC_COLLATE section of the locale file format.

2.2.2. awk script for replace-after

A script has been written in the "awk" language defined in POSIX.2 to implement the "replace-after" construct.

```
BEGIN  {
        comment = "%";
        back[0]= follow[0] = 0
        }

/LC_COLLATE/ { coll=1 }

/END LC_COLLATE/ {
        coll=0;
        for (lnr= 1; lnr; lnr= follow[lnr])
                print cont[lnr]
}

{ if (coll == 0) print $0  ;
  else {
        if ($1 == "copy")   {
                file = $2
                while (getline < file )
                if ( $1 == "LC_COLLATE" ) copy_lc = 1
                else if ( $1 == "END"
                && $2 == "LC_COLLATE" ) copy_lc =0
                else if (copy_lc) {
                        lnr++
                        follow[lnr-1] = lnr
                        back [ lnr ] = lnr-1
                        cont[lnr] = $0
                        symb[ $1 ] = lnr
                }
                close (file )
        }
        else if ($1 == "replace-after")
        { ra=1 ; after = symb [ $2 ] }
        else if ($1 == "replace-end") ra = 0
        else {
                lnr++
                if (ra) follow [ lnr ] = follow [ after ]
                if (ra) back [ follow [ after ] ] = lnr
                follow[after] = lnr
                back [ lnr ] = after
                cont[lnr] = $0
                if ( ra && $1 != comment && $1 != "" )  {
                        old = symb [ $1 ]
                        follow [ back [ old ] ] = follow [ old ]
                        back [ follow [ old ] ] = back [ old ]
                        symb[ $1 ] = lnr
                }
                after = lnr
        }
  }
}
```

2.3. da_DK locale

```
escape_char /
comment_char    %

% Danish language locale for Denmark
% Source: Danish Standards Association
% Contact: Keld Simonsen
% Revision 2.0 1992-07-05

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
collating-element <A-A> from <A><A>
collating-element <a-a> from <a><a>
collating-element <A-a> from <A><a>
copy en_DK
replace-after <CAPITAL>
<CAPITAL>
<BOTH>
<SMALL>
replace-after <q8>
<kk>  <Q>;<SPECIAL>;<SMALL>
replace-after <y8>
<U:>  <Y>;<ACC11>;<CAPITAL>
<u:>  <Y>;<ACC11>;<SMALL>
<U">  <Y>;<ACC12>;<CAPITAL>
<u">  <Y>;<ACC12>;<SMALL>
replace-after <z8>
<AE>  <AE>;<NONE>;<CAPITAL>
<ae>  <AE>;<NONE>;<SMALL>
<AE’> <AE>;<ACUTE>;<CAPITAL>
<ae’> <AE>;<ACUTE>;<SMALL>
<A:>  <AE>;<DIAERESIS>;<CAPITAL>
<a:>  <AE>;<DIAERESIS>;<SMALL>
<A3>  <AE>;<ACC3>;<CAPITAL>
<a3>  <AE>;<ACC3>;<SMALL>
<O//> <O//>;<NONE>;<CAPITAL>
<o//> <O//>;<NONE>;<SMALL>
<O//’>      <O//>;<ACUTE>;<CAPITAL>
<o//’>      <O//>;<ACUTE>;<SMALL>
<O:>  <O//>;<DIAERESIS>;<CAPITAL>
<o:>  <O//>;<DIAERESIS>;<SMALL>
<O">  <O//>;<DOUBLE-ACUTE>;<CAPITAL>
<o">  <O//>;<DOUBLE-ACUTE>;<SMALL>
<AA>  <AA>;<NONE>;<CAPITAL>
<aa>  <AA>;<NONE>;<SMALL>
<A-A> <AA>;<ACC1>;<CAPITAL>
<A-a> <AA>;<ACC1>;<BOTH>
<a-a> <AA>;<ACC1>;<SMALL>
<AA’> <AA>;<ACC3>;<CAPITAL>
<aa’> <AA>;<ACC3>;<SMALL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol         "DKK "
currency_symbol         "kr"
mon_decimal_point       <,>
mon_thousands_sep       <.>
mon_grouping            3;3
positive_sign           ""
negative_sign           <->
int_frac_digits         2
frac_digits             2
p_cs_precedes           1
p_sep_by_space          2
n_cs_precedes           1
n_sep_by_space          2
p_sign_posn             4
n_sign_posn             4
END LC_MONETARY

LC_NUMERIC
decimal_point           <,>
thousands_sep           <.>
grouping                3;3
END LC_NUMERIC

LC_TIME
abday       "s<o//>n";"man";"tir";"ons";/
            "tor";"fre";"l<o//>r"
day         "s<o//>ndag";"mandag";/
            "tirsdag";"onsdag";"torsdag";/
            "fredag";"l<o//>rdag"
abmon       "jan";"feb";"mar";"apr";/
            "maj";"jun";"jul";"aug";/
            "sep";"okt";"nov";"dec"
mon         "januar";"februar";"marts";/
            "april";"maj";"juni";/
            "juli";"august";/
            "september";"oktober";/
            "november";"december"
d_t_fmt     "%a %d %b %Y %T %Z"
d_fmt       "%d-%m-%Y"
t_fmt       "%T"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>1+JjYy<)/>>.*"
noexpr      "<<(>0-Nn<)/>>.*"
END LC_MESSAGES
```

2.4. et_EE locale

```
escape_char  /
comment_char %

% Estonian language locale for Estonia
% Source:
% Estonian Informatics Society
% Contact: Tiit Tammiste
% Date: 1992-04-14

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK

% there are   s and s< different alpha;
% z and z with diacritics after s
% in Estonian

replace_after <s8>
<S<> <S<>;<CARON>;<CAPITAL>
<s<> <S<>;<CARON>;<SMALL>
<Z>  <Z>;<NONE>;<CAPITAL>
<z>  <Z>;<NONE>;<SMALL>
<Z'> <Z>;<ACUTE>;<CAPITAL>
<z'> <Z>;<ACUTE>;<SMALL>
<Z.> <Z>;<DOT>;<CAPITAL>
<z.> <Z>;<DOT>;<SMALL>
<Z<> <Z>;<CARON>;<CAPITAL>
<z<> <Z>;<CARON>;<SMALL>

% there are v=w and
% o?, a:, o:, u: after w in Estonian

replace_after <v>
<W>   <V>;<W>;<NONE>;<CAPITAL>
<w>   <V>;<W>;<NONE>;<SMALL>
<W'>  <V>;<W>;<ACUTE>;<CAPITAL>
<w'>  <V>;<W>;<ACUTE>;<SMALL>
<W!>  <V>;<W>;<GRAVE>;<CAPITAL>
<w!>  <V>;<W>;<GRAVE>;<SMALL>
<W/>> <V>;<W>;<CIRCUMFLEX>;<CAPITAL>
<w/>> <V>;<W>;<CIRCUMFLEX>;<SMALL>
<W-.> <V>;<W>;<MACRON>;<CAPITAL>
<w-.> <V>;<W>;<MACRON>;<SMALL>
<W.>  <V>;<W>;<DOT>;<CAPITAL>
<w.>  <V>;<W>;<DOT>;<SMALL>
<W:>  <V>;<W>;<DIAERESIS>;<CAPITAL>
<w:>  <V>;<W>;<DIAERESIS>;<SMALL>

<O?> <O?>;<TILDE>;<CAPITAL>
<o?> <O?>;<TILDE>;<SMALL>
<A:> <A:>;<DIAERESIS>;<CAPITAL>
<a:> <A:>;<DIAERESIS>;<SMALL>
<O:> <O:>;<DIAERESIS>;<CAPITAL>
<o:> <O:>;<DIAERESIS>;<SMALL>
<U:> <U:>;<DIAERESIS>;<CAPITAL>
<u:> <U:>;<DIAERESIS>;<SMALL>
replace_end
END LC_COLLATE

% Estonia has no own money
% The own money will be in
% operating during the 1992
% This definition is a project

LC_MONETARY
int_curr_symbol         "EEK"
current_symbol          "kr"
mon_decimal_point       <.>
mon_thousands_sep       <NS>
mon_grouping            3;3
positive.sign           ""
negative_sign           <->
int_frac_digits         2
frac_digits             2
p_cs_precedes           1
p_sep_by_space          1
n_cs_precedes           1
n_sep_by_space          1
p_sign_posn             1
n_sign_posn             1
END LC_MONETARY

LC_NUMERIC
decimal_point           <,>
thousands_sep           <NS>
grouping                <3;3>
END LC_NUMERIC

LC_TIME
abday        "P";"E";"T";"K";"N";"R";"L"
day          "p<u:>hap<a:>ev";/
             "esmasp<a:>ev";/
             "teisip<a:>ev";/
             "kolmap<a:>ev";/
             "neljap<a:>ev";/
             "reede";"laup<a:>ev"
abmon        "jaa";"vee";"m<a:>r";/
             "apr";"mai";"jun";/
             "jul";"aug";"sep";/
             "okt";"nov";"det"
mon          "jaanuar";"veebruar";/
             "m<a:>rts";"aprill";/
             "mai";"juuni";/
             "juuli";"august";/
             "september";"oktoober";/
             "november";"detsember"

% Suggested date formats

d_t_fmt       "%a %d %b %Y %T %Z"
```

```
d_fmt           "%d.%m.%Y"
t_fmt           "%T"
am_pm           "";""
t_fmt_ampm      ""
END_LC_TIME

LC_MESSAGES
yesexpr         "<<(>JjYy<)//>.*"
noexpr          "<<(>EeNn<)//>.*"
END LC_MESSAGES
```

2.5. fi_FI locale

```
escape_char /
comment_char    %

% Finnish language locale for Finland
% Source: Finnish Standards
% Contact: Thorbjoern Blomfeldt

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK
% <SMALL> before <CAPITAL> letters
replace-after <CAPITAL>
<SMALL>
<BOTH>
<CAPITAL>
replace-after <y8>
<U:>  <Y>;<ACC11>;<CAPITAL>
<u:>  <Y>;<ACC11>;<SMALL>
replace-after <v-.>
<W>   <V>;<W>;<NONE>;<CAPITAL>
<w>   <V>;<W>;<NONE>;<SMALL>
<W'>  <V>;<W>;<ACUTE>;<CAPITAL>
<w'>  <V>;<W>;<ACUTE>;<SMALL>
<W!>  <V>;<W>;<GRAVE>;<CAPITAL>
<w!>  <V>;<W>;<GRAVE>;<SMALL>
<W/>> <V>;<W>;<CIRCUMFLEX>;<CAPITAL>
<w/>> <V>;<W>;<CIRCUMFLEX>;<SMALL>
<W-.> <V>;<W>;<MACRON>;<CAPITAL>
<w-.> <V>;<W>;<MACRON>;<SMALL>
<W.>  <V>;<W>;<DOT>;<CAPITAL>
<w.>  <V>;<W>;<DOT>;<SMALL>
<W:>  <V>;<W>;<DIAERESIS>;<CAPITAL>
<w:>  <V>;<W>;<DIAERESIS>;<SMALL>
replace-after <z8>
<AA>  <AA>;<NONE>;<CAPITAL>
<aa>  <AA>;<NONE>;<SMALL>
<AA'> <AA>;<ACUTE>;<CAPITAL>
<aa'> <AA>;<ACUTE>;<SMALL>
<A:>  <A:>;<NONE>;<CAPITAL>
<a:>  <A:>;<NONE>;<SMALL>
<AE>  <A:>;<AE>;<CAPITAL>
<ae>  <A:>;<AE>;<SMALL>
<AE'> <A:>;<AE'>;<CAPITAL>
<ae'> <A:>;<AE'>;<SMALL>
<A3>  <A:>;<A3>;<CAPITAL>
<a3>  <A:>;<A3>;<SMALL>
<O:>  <O:>;<NONE>;<CAPITAL>
<o:>  <O:>;<NONE>;<SMALL>
<O//> <O:>;<O//>;<CAPITAL>
<o//> <O:>;<O//>;<SMALL>
<O//'>      <O:>;<O//'>;<CAPITAL>
<o//'>      <O:>;<O//'>;<SMALL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol         "FIM "
currency_symbol         "mk"
mon_decimal_point       <,>
mon_thousands_sep       <.>
mon_grouping            3;3
positive_sign           ""
negative_sign           <->
int_frac_digits         2
frac_digits             2
% int_curr_symbol precedes
% curr_symbol succeeds
p_cs_precedes           0
p_sep_by_space          2
n_cs_precedes           0
n_sep_by_space          2
p_sign_posn             1
n_sign_posn             1
END LC_MONETARY

LC_NUMERIC
decimal_point           <,>
thousands_sep           <.>
grouping                3;0
END LC_NUMERIC

LC_TIME
abday       "su";"ma";"ti";"ke";/
            "to";"pe";"la"
day         "sunnuntai";"maanantai";/
            "tiistai";"keskiviikko";/
            "torstai";"perjantai";/
            "lauantai"
abmon       "tam";"hel";"maa";/
            "huh";"tou";"kes";/
            "hei";"elo";"syy";/
            "lok";"mar";"jou"
mon         "tammikuu";"helmikuu";/
            "maaliskuu";"huhtikuu";/
            "toukokuu";"kes<a:>kuu";/
            "hein<a:>kuu";"elokuu";/
            "syyskuu";"lokakuu";/
            "marraskuu";"joulukuu"
d_t_fmt     "%a %e %B %Y %T"
d_fmt       "%d.%m.%Y"
t_fmt       "%T"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>KkJjYy<)/>>.*"
noexpr      "<<(>NnEe<)/>>.*"
END LC_MESSAGES
```

2.6. fo_FO locale

```
escape_char  /
comment_char %

% Danish locale for the Faroese language
% Source: Danish Standards Association
% Contact: Keld Simonsen
% Revision: 1.8 1992-01-26
%
% Only LC_TIME and LC_MESSAGES are
% specified here, else use the da_DK locale

LC_CTYPE
copy da_DK
END LC_CTYPE

LC_COLLATE
copy da_DK
END LC_COLLATE

LC_MONETARY
copy da_DK
END LC_MONETARY

LC_NUMERIC
copy da_DK
END LC_NUMERIC

LC_TIME
abday       "sun";"m<a'>n";"t<y'>s";"mik";/
            "h<o'>s";"fr<i'>";"ley"
day         "sunnudagur";"m<a'>nadagur";/
            "t<y'>sdagur";/
            "mikudagur";"h<o'>sdagur";/
            "fr<i'>ggjadagur";"leygardagur"
abmon       "jan";"feb";"mar";"apr";/
            "mai";"jun";"jul";"aug";/
            "sep";"okt";"nov";"des"
mon         "januar";"februar";"mars";/
            "apr<i'>l";"mai";"juni";/
            "juli";"august";"september";/
            "oktober";"november";"desember"
d_t_fmt     "%a %d %b %Y %T %Z"
d_fmt       "%d//%m-%Y"
t_fmt       "%T"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>JjYy<)/.*"
noexpr      "<<(>Nn<)/>>.*"
END LC_MESSAGES
```

2.7. is_IS locale

```
comment_char %
escape_char  /
%
% Icelandic Language Locale for Iceland
% source: Stadlarad Islands
% contact: Thorvardur K. Olafsson

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK
replace-after <CAPITAL>
<CAPITAL>
<BOTH>
<SMALL>
replace-after <ae>
<a’>  <a’>;<NONE>;<SMALL>
<A’>  <a’>;<NONE>;<CAPITAL>
replace-after <d8>
<d->  <d->;<NONE>;<SMALL>
<D->  <d->;<NONE>;<CAPITAL>
replace-after <e8>
<e’>  <e’>;<NONE>;<SMALL>
<E’>  <e’>;<NONE>;<CAPITAL>
replace-after <i8>
<i’>  <i’>;<NONE>;<SMALL>
<I’>  <i’>;<NONE>;<CAPITAL>
replace-after <o8>
<o’>  <o’>;<NONE>;<SMALL>
<O’>  <o’>;<NONE>;<CAPITAL>
replace-after <u8>
<u’>  <u’>;<NONE>;<SMALL>
<U’>  <u’>;<NONE>;<CAPITAL>
replace-after <y8>
<y’>  <y’>;<NONE>;<SMALL>
<Y’>  <y’>;<NONE>;<CAPITAL>
replace-after <z8>
<th>  <th>;<NONE>;<SMALL>
<TH>  <th>;<NONE>;<CAPITAL>
<ae>  <ae>;<NONE>;<SMALL>
<AE>  <ae>;<NONE>;<CAPITAL>
<oe>  <ae>;<ACC1>;<SMALL>
<OE>  <ae>;<ACC1>;<CAPITAL>
<o:>  <o:>;<NONE>;<SMALL>
<O:>  <o:>;<NONE>;<CAPITAL>
<o/>  <o:>;<ACC1>;<SMALL>
<O/>  <o:>;<ACC1>;<CAPITAL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol            "ISK "
currency_symbol            "kr"
mon_decimal_point          ","
mon_thousands_sep          "."
mon_grouping               3;3
positive_sign              ""
negative_sign              "-"
% *_frac_* must be checked
int_frac_digits            0
frac_digits                0
p_cs_precedes              0
p_sep_by_space             1
n_cs_precedes              0
n_sep_by_space             1
p_sign_posn                1
n_sign_posn                1
END LC_MONETARY

LC_NUMERIC
decimal_point              ","
thousands_sep              "<SP>"
grouping                   3;3
END LC_NUMERIC

LC_TIME
abday   "sun";"m<a’>n";"<th>ri";"mi<d->";/
        "fim";"f<o:>s";"lau"
day     "sunnudagur";"m<a’>nudagur";/
        "<th>ri<d->judagur";/
        "mi<d->vikudagur";"fimmtudagur";/
        "f<o:>studagur";"laugardagur"
abmon   "jan";"feb";"mar";"apr";"ma<i’>";/
        "j<u’>n";"j<u’>l";"<a’>g<u’>";/
        "sep";"okt";"n<o’>v";"des"
mon     "jan<u’>ar";"febr<u’>ar";"mars";/
        "apr<i’>l";"ma<i’>";"j<u’>n<i’>";/
        "j<u’>l<i’>";"<a’>g<u’>st";/
        "september";"okt<o’>ber";/
        "n<o’>vember";"desember"
d_t_fmt "%d.%m %Y kl. %T"
d_fmt   "%d.%m %Y"
t_fmt   "%T"
am_pm   "";""
t_fmt_ampm ""
END LC_TIME

LC_MESSAGES
yesexpr "<<(>jJyY<)/>>.*"
noexpr  "<<(>nN<)/>>.*"
END LC_MESSAGES
```

2.8. kl_GL locale

```
escape_char /
comment_char %
% Greenlandic language locale for Denmark
% Source: Danish Standards Association
% Contact: Keld Simonsen
% Revision: 1.8 1992-01-26
%
% Only LC_TIME and LC_MESSAGES are
% specified here, else the da_DK locale
% is used.

LC_CTYPE
copy da_DK
END LC_CTYPE

LC_COLLATE
copy da_DK
END LC_COLLATE

LC_MONETARY
copy da_DK
END LC_MONETARY

LC_NUMERIC
copy da_DK
END LC_NUMERIC

LC_TIME
abday       "sab";"ata";"mar";"pin";/
            "sis";"tal";"arf"
day         "sabaat";"ataasinngorneq";/
            "marlunngorneq";/
            "pingasunngorneq";/
            "sisamanngorneq";/
            "tallimanngorneq";/
            "arfininngorneq"
abmon       "jan";"feb";"mar";"apr";/
            "maj";"jun";"jul";"aug";/
            "sep";"okt";"nov";"dec"
mon         "januari";"februari";/
            "martsi";"aprili";"maji";/
            "juni";"juli";"augustusi";/
            "septemberi";"oktoberi";/
            "novemberi";"decemberi"
d_t_fmt     "%a %d %b %Y %T %Z"
d_fmt       "%d %b %Y"
t_fmt       "%T"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>JjYyAa<)/>>.*"
noexpr      "<<(>Nn<)/>>.*"
END LC_MESSAGES
```

2.9. lv_LV locale

```
escape_char /
comment_char %

% Latvian language locale for Latvia
% Source: Latvian working group
% on code tables
% Contact: Imants Metra
% Revision 1.3  1992-09-17

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK

% <CAPITAL> before <SMALL> letters

replace-after <c8>
<C<>   <C<>;<CARON>;<CAPITAL>
<c<>   <C<>;<CARON>;<SMALL>
replace-after <g8>
<G,>   <G,>;<CEDILLA>;<CAPITAL>
<g,>   <G,>;<CEDILLA>;<SMALL>
replace-after <k8>
<K,>   <K,>;<CEDILLA>;<CAPITAL>
<k,>   <K,>;<CEDILLA>;<SMALL>
replace-after <l8>
<L,>   <L,>;<CEDILLA>;<CAPITAL>
<l,>   <L,>;<CEDILLA>;<SMALL>
replace-after <n8>
<N,>   <N,>;<CEDILLA>;<CAPITAL>
<n,>   <N,>;<CEDILLA>;<SMALL>
replace-after <r8>
<R,>   <R,>;<CEDILLA>;<CAPITAL>
<r,>   <R,>;<CEDILLA>;<SMALL>
replace-after <s8>
<S<>   <S<>;<CARON>;<CAPITAL>
<s<>   <S<>;<CARON>;<SMALL>
replace-after <z8>
<Z<>   <Z<>;<CARON>;<CAPITAL>
<z<>   <Z<>;<CARON>;<SMALL>
replace-end
END LC_COLLATE

% Latvia now has a temporary money
% Latvian rouble. In a near future
% the renewal of hard currency
% "lats" is planned. This
% definition is a project for lats.

LC_MONETARY
int_curr_symbol       "LVL "
currency_symbol       "Ls"
mon_decimal_point     <,>
mon_thousands_sep     <NS>
mon_grouping          3;3
positive_sign         ""
negative_sign         <->
int_frac_digits       2
frac_digits           2
p_cs_precedes         1
p_sep_by_space        1
n_cs_precedes         1
n_sep_by_space        1
p_sign_posn           1
n_sign_posn           1
END LC_MONETARY

LC_NUMERIC
decimal_point         <,>
thousands_sep         <NS>
grouping              3;3
END LC_NUMERIC

LC_TIME
abday        "Sv";"P ";"O ";"T ";"C ";"Pk";"S "
day          "sv<e->tdiena";"pirmdiena";/
             "otrdiena";"tre<s<>diena";/
             "ceturtdiena";/
             "piektdiena";"sestdiena"
abmon        "jan";"feb";"mar";"apr";/
             "mai";"j<u->n";"j<u->l";"aug";/
             "sep";"okt";"nov";"dec"
mon          "janv<a->ris";"febru<a->ris"/
             "marts";"apr<i->lis";/
             "maijs";"j<u->nijs";/
             "j<u->lijs";"augusts";/
             "septembris";"oktobris";/
             "novembris";"decembris"
d_t_fmt      "%A, %Y. gada %e. %B, plkst. %H.%M"
d_fmt        "%d.%m.%y."
t_fmt        "%T"
am_pm        "";""
t_fmt_ampm   ""
END LC_TIME

LC_MESSAGES
yesexpr    "<<(>JjYy<)//>.*"
noexpr     "<<(>Nn<)//>.*"
END LC_MESSAGES
```

2.10. lt_LT locale

```
escape_char /
comment_char %

% Lithuanian language locale for Lithuania
% Created on January 30, 1992
% Revised on April 10, 1992
% Source: Lithuanian Computer Society
% Contact: Evaldas Kulbokas

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK
replace-after <i8>
<Y> <Y>;<NONE>;<CAPITAL>
<y> <Y>;<NONE>;<SMALL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol     "LTL "
currency_symbol     "Lt"
mon_decimal_point   <,>
mon_thousands_sep   <.>
mon_grouping        3;3
positive_sign       ""
negative_sign       <->
int_frac_digits     2
frac_digits         2
p_cs_precedes       0
p_sep_by_space      1
n_cs_precedes       0
n_sep_by_space      1
p_sign_posn         1
n_sign_posn         1
END LC_MONETARY

LC_NUMERIC
decimal_point       <,>
thousands_sep       <.>
grouping            3;3
END LC_NUMERIC

LC_TIME
abday     "sek";"pir";"ant";"tre";/
          "ket";"pen";"<s<>e<s<>"
day       "sekmadienis";"pirmadienis";/
          "antradienis";"tre<c<>iadienis";/
          "ketvirtadienis";"penktadienis";/
          "<s<>e<s<>tadienis"
abmon     "sau";"vas";"kov";/
          "bal";"geg";"bir";/
          "lie";"rgp";"rgs";/
          "spa";"lap";"grd"
mon       "sausio";"vasario";"kovo";/
          "baland<z<>io";/
          "gegu<z<><e.>s";/
          "bir<z<>elio";"liepos";/
          "rugpj<u-><c<>io";/
          "rugs<e.>jo";"spalio";/
          "lapkri<c<>io";"gruod<z<>io"
d_t_fmt   "%Y m. %B %d d. %T"
d_fmt     "%Y.%m.%d"
t_fmt     "%T"
am_pm     "";""
t_fmt_ampm ""
END LC_TIME

LC_MESSAGES
yesexpr    "<<(>TtYy<)//>.*"
noexpr     "<<(>Nn<)//>.*"
END LC_MESSAGES
```

2.11. no_NO locale

```
escape_char /
comment_char    %

% Norwegian language locale for Norway
% Source: Norsk Standardiseringsforbund
% contact: Kolbjoern Aamboe
% Revision 1.0

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK
% <CAPITAL> before <SMALL> letters
replace-after <CAPITAL>
<CAPITAL>
<BOTH>
<SMALL>
collating-element <A-A> from <A><A>
collating-element <a-a> from <a><a>
collating-element <A-a> from <A><a>
copy en_DK
% <CAPITAL> before <SMALL> letters
replace-after <CAPITAL>
<CAPITAL>
<BOTH>
<SMALL>
replace-after <y8>
<U:>  <Y>;<ACC11>;<CAPITAL>
<u:>  <Y>;<ACC11>;<SMALL>
<U">  <Y>;<ACC12>;<CAPITAL>
<u">  <Y>;<ACC12>;<SMALL>
replace-after <z8>
<AE>  <AE>;<NONE>;<CAPITAL>
<ae>  <AE>;<NONE>;<SMALL>
<AE'> <AE>;<ACUTE>;<CAPITAL>
<ae'> <AE>;<ACUTE>;<SMALL>
<A3>  <AE>;<ACC3>;<CAPITAL>
<a3>  <AE>;<ACC3>;<SMALL>
<A:>  <AE>;<ACC8>;<CAPITAL>
<a:>  <AE>;<ACC8>;<SMALL>
<O//> <O//>;<NONE>;<CAPITAL>
<o//> <O//>;<NONE>;<SMALL>
<O//'>      <O//>;<ACUTE>;<CAPITAL>
<o//'>      <O//>;<ACUTE>;<SMALL>
<O:>  <O//>;<DIAERESIS>;<CAPITAL>
<o:>  <O//>;<DIAERESIS>;<SMALL>
<O">  <O//>;<DOUBLE-ACUTE>;<CAPITAL>
<o">  <O//>;<DOUBLE-ACUTE>;<SMALL>
<AA>  <AA>;<NONE>;<CAPITAL>
<aa>  <AA>;<NONE>;<SMALL>
<A-A> <AA>;<ACC1>;<CAPITAL>
<A-a> <AA>;<ACC1>;<BOTH>
<a-a> <AA>;<ACC1>;<SMALL>
<AA'> <AA>;<ACC3>;<CAPITAL>
<aa'> <AA>;<ACC3>;<SMALL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol         "NOK "
currency_symbol         "kr"
mon_decimal_point       <,>
mon_thousands_sep       <.>
mon_grouping            3;3
positive_sign           ""
negative_sign           <->
int_frac_digits         2
frac_digits             2
p_cs_precedes           1
p_sep_by_space          0
n_cs_precedes           1
n_sep_by_space          0
p_sign_posn             4
n_sign_posn             4
END LC_MONETARY

LC_NUMERIC
decimal_point           <,>
thousands_sep           <.>
grouping                3;3
END LC_NUMERIC

LC_TIME
abday       "s<o//>n";"man";"tir";"ons";/
            "tor";"fre";"l<o//>r"
day         "s<o//>ndag";"mandag";/
            "tirsdag";"onsdag";/
            "torsdag";"fredag";"l<o//>rdag"
abmon       "jan";"feb";"mar";"apr";/
            "mai";"jun";"jul";"aug";/
            "sep";"okt";"nov";"des"
mon         "januar";"februar";"mars";/
            "april";"mai";"juni";/
            "juli";"august";"september";/
            "oktober";"november";"desember"
% usual date representation
d_t_fmt     "%a %d-%m-%Y %T %Z"
d_fmt       "%d-%m-%Y"
t_fmt       "%T"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>JjYy<)/>>.*"
noexpr      "<<(>Nn<)/>>.*"
END LC_MESSAGES
```

2.12. sv_FI locale

```
comment_char %
escape_char  /
%
% Swedish Language Locale for Finland
% Source: Finnish Standards
% Contact: Thorbjoern Blomfeldt

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy sv_SE
END LC_COLLATE

LC_MESSAGES
copy sv_SE
END LC_MESSAGES

LC_MONETARY
copy fi_FI
END LC_MONETARY

LC_NUMERIC
copy fi_FI
END LC_NUMERIC

LC_TIME
copy sv_SE
END LC_TIME
```

2.13. sv_SE locale

```
escape_char /
comment_char    %

% Swedish language locale for Sweden
% source: SIS-ITS
% contact: Peter Svanberg

LC_CTYPE
copy en_DK
END LC_CTYPE

LC_COLLATE
copy en_DK
% <SMALL> before <CAPITAL> letters
replace-after <CAPITAL>
<SMALL>
<BOTH>
<CAPITAL>
replace-after <y8>
<U:>  <Y>;<ACC11>;<CAPITAL>
<u:>  <Y>;<ACC11>;<SMALL>
replace-after <v-.>
<W>   <V>;<W>;<NONE>;<CAPITAL>
<w>   <V>;<W>;<NONE>;<SMALL>
<W'>  <V>;<W>;<ACUTE>;<CAPITAL>
<w'>  <V>;<W>;<ACUTE>;<SMALL>
<W!>  <V>;<W>;<GRAVE>;<CAPITAL>
<w!>  <V>;<W>;<GRAVE>;<SMALL>
<W/>> <V>;<W>;<CIRCUMFLEX>;<CAPITAL>
<w/>> <V>;<W>;<CIRCUMFLEX>;<SMALL>
<W-.> <V>;<W>;<MACRON>;<CAPITAL>
<w-.> <V>;<W>;<MACRON>;<SMALL>
<W.>  <V>;<W>;<DOT>;<CAPITAL>
<w.>  <V>;<W>;<DOT>;<SMALL>
<W:>  <V>;<W>;<DIAERESIS>;<CAPITAL>
<w:>  <V>;<W>;<DIAERESIS>;<SMALL>
replace-after <z8>
<AA>  <AA>;<NONE>;<CAPITAL>
<aa>  <AA>;<NONE>;<SMALL>
<AA'> <AA>;<ACUTE>;<CAPITAL>
<aa'> <AA>;<ACUTE>;<SMALL>
<A:>  <A:>;<NONE>;<CAPITAL>
<a:>  <A:>;<NONE>;<SMALL>
<AE>  <A:>;<AE>;<CAPITAL>
<ae>  <A:>;<AE>;<SMALL>
<AE'> <A:>;<AE'>;<CAPITAL>
<ae'> <A:>;<AE'>;<SMALL>
<A3>  <A:>;<A3>;<CAPITAL>
<a3>  <A:>;<A3>;<SMALL>
<O:>  <O:>;<NONE>;<CAPITAL>
<o:>  <O:>;<NONE>;<SMALL>
<O//> <O:>;<O//>;<CAPITAL>
<o//> <O:>;<O//>;<SMALL>
<O//'>      <O:>;<O//'>;<CAPITAL>
<o//'>      <O:>;<O//'>;<SMALL>
replace-end
END LC_COLLATE

LC_MONETARY
int_curr_symbol         "SEK "
currency_symbol         "kr"
mon_decimal_point       <,>
mon_thousands_sep       <SP>
mon_grouping            3;0
positive_sign           ""
negative_sign           <->
int_frac_digits         2
frac_digits             2
p_cs_precedes           0
p_sep_by_space          1
n_cs_precedes           0
n_sep_by_space          1
p_sign_posn             1
n_sign_posn             1
END LC_MONETARY

LC_NUMERIC
decimal_point           <,>
thousands_sep           <SP>
grouping                3;0
END LC_NUMERIC

LC_TIME
abday       "s<o:>n";"m<aa>n";"tis";"ons";/
            "tor";"fre";"l<o:>r"
day         "s<o:>ndag";"m<aa>ndag";/
            "tisdag";"onsdag";/
            "torsdag";"fredag";"l<o:>rdag"
abmon       "jan";"feb";"mar";"apr";/
            "maj";"jun";"jul";"aug";/
            "sep";"okt";"nov";"dec"
mon         "januari";"februari";"mars";/
            "april";"maj";"juni";/
            "juli";"augusti";"september";/
            "oktober";"november";"december"
d_t_fmt     "%a %e %b %Y %H.%M.%S"
d_fmt       "%Y-%m-%d"
t_fmt       "%H.%M.%S"
am_pm       "";""
t_fmt_ampm  ""
END LC_TIME

LC_MESSAGES
yesexpr     "<<(>JjYy<)/>>.*"
noexpr      "<<(>Nn<)/>>.*"
END LC_MESSAGES
```

3. Format of Character Set Codetables

The character menmonics hav been used to table a number of coded character sets. The coded character set names are taken if possible from the official ISO registration description in the ISO 2375 (ECMA) register, or with a number like the code page number - or with an indication of the language or country it is being used for - using the country designators of ISO 3166. For the character sets in the ECMA register, their ISO registration number is also given (as ISO-IR-xxx). Often the ISO registation number does not cover all the codes of a character set in use, but for instance only the graphical characters, where another ISO registration number covers the control characters; in the case of the 8-bit character sets the ISO registration only covers the upper graphical characters (GR). The ISO registration number is here taken to indicate the full coded character set including control characters and lower half of the graphical characters, normally ISO 6429 and ASCII, respectively.

The ISO definition of the term "coded character set" is as follows: "A set of unambiguous rules that establishes a character set and the one-to-one relationship between the characters of the set and their coded representation." and this definition may be subject to different interpretations. This memo does not put further restrictions on the term of "coded character set" than the following: "A coded character set is a set of rules that unambiguously and completely determines which sequence of characters, if any, is represented by each possible sequence of n-bit bytes for a certain value of n." This implies that e.g. a coded character set extended with one or more other coded character sets by means of the extension techniques of ISO 2022 constitutes a coded character set in its own right. In this memo the term "charset" is used to refer to the above interpretation of the ISO term "coded character set".

A special problem is, if two characters of two different coded character sets with the same descriptive name, or depicted by what looks like the same graphic symbol, or with the same historical origin, really are to be regarded as the same character or not. This problem has been studied in great detail in the development efforts that have resulted in ISO DIS 10646 and Unicode (under the heading "character unification"). As much as possible such results have been used in the construction of the code tables of this section.

3.1. Charset Naming

The coded character set names are given in ISO 646 invariant subset (83 characters, where a space in the name is replaced with an underline character; sometimes a hyphen is also used instead of a blank, or the blank is eliminated when practice exist). Case is not significant in the charset names.

3.2. Code Table Format

The following code tables are given in a simple format to faciliate use of this text as program input. Programs and routines written in C to handle these tables are freely available from the author of this memo. Keywords are signified with the character "&" as the first character, to distinquish them from ordinary data. Numbers may be given in decimal, hexadecimal or octal notation; hexadecimal numbers are given with an "x" as the first character, and octal numbers has an "o" as the first character.

The following keywords are used:

"&charset" has one parameter defining the name of the character set. This is required for every character set.

"&alias" has one parameter defining a possible alternate name for the character set. This is optional.

"&g0esc", "&g1esc", "&g2esc", "&g3esc", "&c0esc", "&c1esc" has one parameter indicating the string of octets used to define the character set as the G0, G1, G2, G3, C0 or C1 set respectively, according to ISO 2022 (11). The string is to be preceded by an ESC character. It is only the relevant parts of the table, which can be used with the definition; the charset is often coded with both graphical and control character sets. If the coded character set is a 96-character set, it is tabled with the relevant GL set (normally ISO-IR-6) and with ISO 6429 as C0 and C1 (12). If it is a 94-character set, it is tabled with the C0 set of ISO 6429. If it is a double-octet coded character set, it is tabled without control character sets and accompanying one-octet coded character sets, and the two-octet code is tabled as a G0 set.

"&bits" has one parameter indicating the number of bits to represent the charset. This is optional and 8 bits is the default.

"&code" has one parameter indicating the byte number allocated to the following character mnemonic. After the "&code" specification the characters are listed with their mnemonic in ascending order. A character mnemonic of "??" indicates that the position is unused. A character mnemonic of "__" indicates that the character set is not completely defined with the specifications in this memo.

"&code2" has 2 parameters specifying the row and column in certain 16-bit character sets. The value 32 must be added to obtain the first and second byte respectively. Mnemonics can be specified after the "&code2" specification as mentioned for the "&code" specification.

"&codex" has 5 parameters, specifying the character set prefix string, the start row number, the end row number, the start column number and the end column number respectively. This is equivalent to specifying a series of mnemonics of the form "nrrcc" where "n" is the character set name prefix string, "rr" is the row number running from the specified start row number to the end row number, and "cc" is the column number running from the specified start column number to the end column number. The thereby created series mnemonics are allocated to code positions which are added 32 to the row and column numbers to get the row and column octet.

"&duplicate" has a special meaning indicating that a position is being used for more than one character. This is an ugly convention but it is a sad fact of life that same code in one coded character set can mean different characters. "&duplicate" takes two parameters - the first is the code to be duplicated, the other is the new mnemonic.

"&rem" is followed by text to explain something in the table to a human reader. All lines in such a remark has to start with this keyword.

"&comb2" specifies a combination of two characters which signifies a third character. All characters in the specification are given by their mnemonic. The two combining characters must be specified previously in the code table. The first combining character is specified as the first character after the keyword, and then the following pairs of characters are the second combining character and the result, respectively. The specification can be repeated, terminated by an occurance of a keyword.

4. Usage of Coded Character Sets

4.1. Recommended Coded Character Sets for the Nordic countries

4.2. Common Coded Character Sets for the Nordic countries

The character sets used in the Nordic countries are the following:

ANSI_X3.4-1968
SEN_850200_B
SEN_850200_C
NS_4551-1
NS_4551-2
ISO_8859-1:1987
ISO_8859-4:1988
la6
ISO_6937-2-add
latin-lap
DS_2089
hp-roman8
macintosh
macintosh-is
IBM037
IBM038
IBM277
IBM278
IBM437
IBM500
IBM819
IBM850
IBM861
IBM865
IBM871
EBCDIC-DK-NO
EBCDIC-DK-NO-A
EBCDIC-FI-SE
EBCDIC-FI-SE-A
EBCDIC-US
FRISS

In Denmark the following character sets are in use:

ISO_8859-1:1987 HP-roman8 MacIntosh IBM850 IBM865 EBCDIC-DK-NO T.61 DEC-MCS DS2089 ASCII

In Iceland the following character sets are in use:

ISO_8859-1:1987 HP-roman8 MacIntosh-IS IBM850 IBM861 EBCDIC-IS

```
&charset ANSI_X3.4-1968
&rem source: ECMA registry
&alias iso-ir-6
&alias ANSI_X3.4-1986
&alias ISO_646.irv:1991
&g0esc x2842 &g1esc x2942 &g2esc x2a42 &g3esc x2b42
&alias ASCII
&alias ISO646-US
&alias US-ASCII
&alias us
&alias IBM367
&alias cp367
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT

&charset SEN_850200_B
&rem source: ECMA registry
&alias iso-ir-10
&alias FI
&alias ISO646-FI
&alias ISO646-SE
&alias se
&g0esc x2847 &g1esc x2947 &g2esc x2a47 &g3esc x2b47
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb Cu % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z A: O: AA '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z a: o: aa '- DT

&charset SEN_850200_C
&rem source: ECMA registry
&alias iso-ir-11
&alias ISO646-SE2
&g0esc x2848 &g1esc x2948 &g2esc x2a48 &g3esc x2b48
&alias se2
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb Cu % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
E' A B C D E F G H I J K L M N O P Q R S T U V W X Y Z A: O: AA U: _
e' a b c d e f g h i j k l m n o p q r s t u v w x y z a: o: aa u: DT

&charset NS_4551-1
&rem source: ECMA registry
&alias iso-ir-60
&alias ISO646-NO
&g0esc x2860 &g1esc x2960 &g2esc x2a60 &g3esc x2b60
&alias no
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
```

```
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z AE O/ AA '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z ae o/ aa '- DT

&charset NS_4551-2
&rem source: ECMA registry
&alias ISO646-NO2
&alias iso-ir-61
&g0esc x2861 &g1esc x2961 &g2esc x2a61 &g3esc x2b61
&alias no2
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " PI DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z AE O/ AA '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z ae o/ aa !! DT

&charset ISO_8859-1:1987
&rem source: ECMA registry
&alias iso-ir-100
&g1esc x2d41 &g2esc x2e41 &g3esc x2f41
&alias ISO_8859-1
&alias ISO-8859-1
&alias latin1
&alias l1
&alias IBM819
&alias CP819
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
NS !I Ct Pd Cu Ye BB SE ': Co -a << NO -- Rg '-
DG +- 2S 3S '' My PI .M ', 1S -o >> 14 12 34 ?I
A! A' A> A? A: AA AE C, E! E' E> E: I! I' I> I:
D- N? O! O' O> O? O: *X O/ U! U' U> U: Y' TH ss
a! a' a> a? a: aa ae c, e! e' e> e: i! i' i> i:
d- n? o! o' o> o? o: -: o/ u! u' u> u: y' th y:

&charset ISO_8859-4:1988
&rem source: ECMA registry
&alias iso-ir-110
&g1esc x2d44 &g2esc x2e44 &g3esc x2f44
&alias ISO_8859-4
&alias ISO-8859-4
&alias latin4
&alias l4
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
```

```
NS A; kk R, Cu I? L, SE ': S< E- G, T/ -- Z< '-
DG a; '; r, '' i? l, '< ', s< e- g, t/ NG z< ng
A- A' A> A? A: AA AE I; C< E' E; E: E. I' I> I-
D/ N, O- K, O> O? O: *X O/ U; U' U> U: U? U- ss
a- a' a> a? a: aa ae i; c< e' e; e: e. i' i> i-
d/ n, o- k, o> o? o: -: o/ u; u' u> u: u? u- '.

&charset ISO_6937-2-add
&rem source: ECMA registry and ISO 6937-2:1983
&alias iso-ir-142
&g1esc x2d4a &g2esc x2e4a &g3esc x2f4a
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb Cu % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
NS !I Ct Pd DO Ye ?? SE ?? '6 "6 << <- -! -> -v
DG +- 2S 3S *X My PI .M -: '9 "9 >> 14 12 34 ?I
?? "! "' "> "? "- "( ". ": ?? "0 ", "_ "" "; "<
-M 1S Rg Co TM Md NO BB ?? ?? ?? ?? 18 38 58 78
Om AE D- -a H/ ?? IJ L. L/ O/ OE -o TH T/ NG 'n
kk ae d/ d- h/ i. ij l. l/ o/ oe ss th t/ ng --
&comb2 "' SP '' a a' A A' e e' E E' i i' I I' o o' O O' u u' U U'
                y y' Y Y' c c' C C' l l' L L' n n' N N' r r' R R'
                s s' S S' z z' Z Z'
&comb2 "( SP '( a a( A A( g g( G G(                     u u( U U(
&comb2 "< SP '< c c< C C< d d< D D< e e< E E< l l< L L< n n< N N<
                r r< R R< s s< S S< t t< T T< z z< Z Z<
&comb2 ", SP ', c c, C C, g g, G G, k k, K K, l l, L L, n n, N N,
                r r, R R, s s, S S, t t, T T,
&comb2 ">       a a> A A> e e> E E> i i> I I> o o> O O> u u> U U>
                y y> Y Y> c c> C C> g g> G G> h h> H H> j j> J J>
                s s> S S> w w> W W>
&comb2 ": SP ': a a: A A: e e: E E: i i: I I: o o: O O: u u: U U:
                y y: Y Y:
&comb2 ". SP '. c c. C C. e e. E E. g g. G G.      I I. z z. Z Z.
&comb2 "" SP '"                               o o" O O" u u" U U"
&comb2 "!       a a! A A! e e! E E! i i! I I! o o! O O! u u! U U!
&comb2 "- SP '- a a- A A- e e- E E- i i- I I- o o- O O- u u- U U-
&comb2 "; SP '; a a; A A; e e; E E; i i; I I; u u; U U;
&comb2 "0 SP '0 a aa A AA                               u u0 U U0
&comb2 "?       a a? A A? n n? N N? i i? I I? o o? O O? u u? U U?

&charset latin6
&rem source: ECMA registry
&alias iso-ir-157
&alias l6
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
```

```
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
NS A; E- G, I- I? K, L, N' R, S< T/ Z< -- kk NG
d/ a; e- g, i- i? k, l, n' r, s< t/ z< SE ss ng
A- A' A> A? A: AA AE I; C< E' E; E: E. I' I> I:
D/ N, O- O' O> O? O: U? O/ U; U' U> U: Y' TH U-
a- a' a> a? a: aa ae i; c< e' e; e: e. i' i> i:
d- n, o- o' o> o? o: u? o/ u; u' u> u: y' th u-

&charset latin-lap
&rem source: ECMA registry
&alias lap
&alias iso-ir-158
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
'' ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
;S ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
A( A> A! A- A1 A7 A3 E( E! E- G/ G< K< O( O! O;
O1 U- ED EZ ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
a( a> a! a- a1 a7 a3 e( e! e- g/ g< k< o( o! o;
o1 u- ed ez ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??

&charset DS_2089
&rem source: Danish Standard, DS 2089, February 1974
&alias DS2089
&alias ISO646-DK
&alias dk
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z AE O/ AA '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z ae o/ aa '? DT

&charset hp-roman8
&rem source: LaserJet IIP Printer User's Manual,
&rem HP part no 33471-90901, Hewlet-Packard, June 1989.
&alias roman8
&alias r8
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
PA HO BH NH IN NL SA ES HS HJ VS PD PU RI S2 S3
DC P1 P2 TS CC MW SG EG SS GC SC CI ST OC PM AC
NS A! A> E! E> E: I> I: '' 1" AN ': ?1 U! U> Li
'- Y' y' DG C, c, N? n? !I ?I Cu Pd Ye SE Fl Ct
a> e> o> u> a' e' o' u' a! e! o! u! a: e: o: u:
AA i> O/ AE aa i' o/ ae A: i! O: U: E' i: ss O>
A' A? a? D- d- I' I! O' O! O? o? S< s< U' Y: y:
```

```
TH th .M My PI 34 -M 14 12 -a -o << fS >> +- ??

&charset macintosh
&rem source: The Unicode Standard ver 1.0, ISBN 0-201-56788-1, Oct 1991
&alias mac
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ’ ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> ’> _
’! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) ’? DT
A: AA C, E’ N? O: U: a’ a! a> a: a? aa c, e’ e!
e> e: i’ i! i> i: n? o’ o! o> o: o? u’ u! u> u:
/- DG Ct Pd SE Sb PI ss Rg Co TM ’’ ’: != AE O/
00 +- =< >= Ye My dP +Z *P p* In -a -o W* ae o/
?I !I NO RT Fl ?2 D* << >> .3 NS A! A? O? OE oe
-M -N "6 "9 ’6 ’9 -: Db y: Y: /f Cu <1 >1 fi fl
/= .M .9 :9 %0 A> E> A’ E: E! I’ I> I: I! O’ O>
?? O! U’ U> U! i. ?? ?? ’- ’( ’. ’0 ’, ’" ’; ’<

&charset IBM037
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias cp037
&alias ebcdic-cp-us
&alias ebcdic-cp-ca
&alias ebcdic-cp-wt
&alias ebcdic-cp-nl
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP NS a> a: a! a’ a? aa c, n? Ct .  <  (  +  !!
&  e’ e> e: e! i’ i> i: i! ss !  DO *  )  ;  NO
-  /  A> A: A! A’ A? AA C, N? BB ,  %  _  >  ?
o/ E’ E> E: E! I’ I> I: I! ’! :  Nb At ’  =  "
O/ a  b  c  d  e  f  g  h  i  << >> d- y’ th +-
DG j  k  l  m  n  o  p  q  r  -a -o ae ’, AE Cu
My ’? s  t  u  v  w  x  y  z  !I ?I D- Y’ TH Rg
’> Pd Ye .M Co SE PI 14 12 34 <( )> ’- ’: ’’ *X
(! A  B  C  D  E  F  G  H  I  -- o> o: o! o’ o?
!) J  K  L  M  N  O  P  Q  R  1S u> u: u! u’ y:
// -: S  T  U  V  W  X  Y  Z  2S O> O: O! O’ O?
0  1  2  3  4  5  6  7  8  9  3S U> U: U! U’ DT

&charset IBM038
&rem source: IBM 3174 Character Set Ref, GA27-3831-02, March 1990
&alias EBCDIC-INT
&alias cp038
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? <( .  <  (  +  !
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? )> DO *  )  ;  ’>
-  /  ?? ?? ?? ?? ?? ?? ?? ?? BB ,  %  _  >  ?
```

```
?? ?? ?? ?? ?? ?? ?? ?? ?? ’! :  Nb At ’  =  "
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
?? ’? s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
(! A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
!) J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
// ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT

&charset IBM277
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias EBCDIC-CP-DK
&alias EBCDIC-CP-NO
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP NS a> a: a! a’ a? !) c, n? Nb .  <  (  +  !
&  e’ e> e: e! i’ i> i: i! ss Cu AA *  )  ;  ’>
-  /  A> A: A! A’ A? DO C, N? o/ ,  %  _  >  ?
BB E’ E> E: E! I’ I> I: I! ’! :  AE O/ ’  =  "
At a  b  c  d  e  f  g  h  i  << >> d- y’ th +-
DG j  k  l  m  n  o  p  q  r  -a -o (! ’, <( )>
My u: s  t  u  v  w  x  y  z  !I ?I D- Y’ TH Rg
Ct Pd Ye .M Co SE PI 14 12 34 NO !! ’- ’: ’’ *X
ae A  B  C  D  E  F  G  H  I  -- o> o: o! o’ o?
aa J  K  L  M  N  O  P  Q  R  1S u> ’? u! u’ y:
// -: S  T  U  V  W  X  Y  Z  2S O> O: O! O’ O?
0  1  2  3  4  5  6  7  8  9  3S U> U: U! U’ DT

&charset IBM278
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias CP278
&alias ebcdic-cp-fi
&alias ebcdic-cp-se
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP NS a> (! a! a’ a? !) c, n? SE .  <  (  +  !
&  ’! e> e: e! i’ i> i: i! ss Cu AA *  )  ;  ’>
-  /  A> Nb A! A’ A? DO C, N? o: ,  %  _  >  ?
o/ E’ E> E: E! I’ I> I: I! e’ :  A: O: ’  =  "
O/ a  b  c  d  e  f  g  h  i  << >> d- y’ th +-
DG j  k  l  m  n  o  p  q  r  -a -o ae ’, AE )>
My u: s  t  u  v  w  x  y  z  !I ?I D- Y’ TH Rg
Ct Pd Ye .M Co <( PI 14 12 34 NO !! ’- ’: ’’ *X
a: A  B  C  D  E  F  G  H  I  -- o> BB o! o’ o?
aa J  K  L  M  N  O  P  Q  R  1S u> ’? u! u’ y:
// -: S  T  U  V  W  X  Y  Z  2S O> At O! O’ O?
0  1  2  3  4  5  6  7  8  9  3S U> U: U! U’ DT

&charset IBM437
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
```

```
&alias cp437
&alias 437
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
C, u: e' a> a: a! aa c, e> e: e! i: i> i! A: AA
E' ae AE o> o: o! u> u! y: O: U: Ct Pd Ye Pt Fl
a' i' o' u' n? N? -a -o ?I NI NO 12 14 !I << >>
.S :S ?S vv vl vL Vl Dl dL VL VV LD UL Ul uL dl
ur uh dh vr hh vh vR Vr UR DR UH DH VR HH VH uH
Uh dH Dh Ur uR dR Dr Vh vH ul dr FB LB lB RB TB
a* b* G* p* S* s* m* t* F* H* W* d* 00 /0 e* (U
=3 +- >= =< Iu Il -: ?2 Ob .M Sb RT nS 2S fS NS

&charset IBM500
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias CP500
&alias ebcdic-cp-be
&alias ebcdic-cp-ch
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP NS a> a: a! a' a? aa c, n? <( .  <  (  +  !
&  e' e> e: e! i' i> i: i! ss )> DO *  )  ;  '>
-  /  A> A: A! A' A? AA C, N? BB ,  %  _  >  ?
o/ E' E> E: E! I' I> I: I! '! :  Nb At '  =  "
O/ a  b  c  d  e  f  g  h  i  << >> d- y' th +-
DG j  k  l  m  n  o  p  q  r  -a -o ae ', AE Cu
My '? s  t  u  v  w  x  y  z  !I ?I D- Y' TH Rg
Ct Pd Ye .M Co SE PI 14 12 34 NO !! '- ': '' *X
(! A  B  C  D  E  F  G  H  I  -- o> o: o! o' o?
!) J  K  L  M  N  O  P  Q  R  1S u> u: u! u' y:
// -: S  T  U  V  W  X  Y  Z  2S O> O: O! O' O?
0  1  2  3  4  5  6  7  8  9  3S U> U: U! U' DT

&charset IBM850
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias cp850
&alias 850
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
C, u: e' a> a: a! aa c, e> e: e! i: i> i! A: AA
E' ae AE o> o: o! u> u! y: O: U: o/ Pd O/ *X Fl
a' i' o' u' n? N? -a -o ?I Rg NO 12 14 !I << >>
.S :S ?S vv vl A' A> A! Co VL VV LD UL Ct Ye dl
ur uh dh vr hh vh a? A? UR DR UH DH VR HH VH Cu
d- D- E> E: E! i. I' I> I: ul dr FB LB BB I! TB
O' ss O> O! o? O? m* TH th U' U> U! y' Y' -M ''
```

```
-- +- == 34 PI SE -: '; DG ': '. 1S 3S 2S fS NS

&charset IBM861
&rem source: IBM NLS RM Vol2 SE09-8002-01, March 1990
&alias cp861
&alias 861
&alias cp-is
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
C, u: e' a> a: a! aa c, e> e: e! D- d- TH A: AA
E' ae AE o> o: th u> Y' y' O: U: o/ Pd O/ Pt O!
a' i' o' u' A' I' O' U' ?I ?? NO 12 14 !I << >>
.S :S ?S vv vl vL Vl Dl dL VL VV LD UL Ul uL dl
ur uh dh vr hh vh vR Vr UR DR UH DH VR HH VH uH
Uh dH Dh Ur uR dR Dr Vh vH ul dr FB LB lB RB TB
a* b* G* p* S* s* m* t* F* H* W* d* 00 /0 e* (U
=3 +- >= =< Iu Il -: ?2 Ob .M Sb RT nS 2S fS NS

&charset IBM865
&rem source: IBM DOS 3.3 Ref (Abridged), 94X9575 (Feb 1987)
&alias cp865
&alias 865
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
SP ! " Nb DO % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
At A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <( // )> '> _
'! a b c d e f g h i j k l m n o p q r s t u v w x y z (! !! !) '? DT
C, u: e' a> a: a! aa c, e> e: e! i: i> i! A: AA
E' ae AE o> o: o! u> u! y: O: U: o/ Pd O/ Pt Fl
a' i' o' u' n? N? -a -o ?I NI NO 12 14 !I << >>
.S :S ?S vv vl vL Vl Dl dL VL VV LD UL Ul uL dl
ur uh dh vr hh vh vR Vr UR DR UH DH VR HH VH uH
Uh dH Dh Ur uR dR Dr Vh vH ul dr FB LB lB RB TB
a* b* G* p* S* s* m* t* F* H* W* d* 00 /0 e* (U
=3 +- >= =< Iu Il -: ?2 Ob .M Sb RT nS 2S fS NS

&charset EBCDIC-DK-NO
&rem source: IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? Nb .  <  (  +  !
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? Cu AA *  )  ;  '>
-  /  ?? ?? ?? ?? ?? ?? ?? ?? o/ ,  %  _  >  ?
?? ?? ?? ?? ?? ?? ?? ?? ?? '! :  AE O/ '  =  "
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
?? u: s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
ae A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
```

```
aa J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
// ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT

&charset EBCDIC-DK-NO-A
&rem source: IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? o/ .  <  (  +  !!
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? aa AA *  )  ;  NO
-  /  ?? ?? ?? ?? ?? ?? ?? ?? BB ,  %  _  >  ?
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? :  AE O/ '  =  ae
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
?? ?? s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
?? J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
?? ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT

&charset EBCDIC-FI-SE
&rem source: IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? SE .  <  (  +  !
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? Cu AA *  )  ;  '>
-  /  ?? ?? ?? ?? ?? ?? ?? ?? o: ,  %  _  >  ?
?? ?? ?? ?? ?? ?? ?? ?? ?? e' :  A: O: '  =  "
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
?? u: s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
a: A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
aa J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
E' ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT

&charset EBCDIC-FI-SE-A
&rem source: IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? o: .  <  (  +  !!
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? aa AA *  )  ;  NO
-  /  ?? ?? ?? ?? ?? ?? ?? ?? BB ,  %  _  >  ?
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? :  A: O: '  =  a:
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
```

```
?? ?? s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
?? J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
?? ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT

&charset EBCDIC-US
&rem source: IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987
&code 0
NU SH SX EX ET EQ AK BL BS HT LF VT FF CR SO SI
DL D1 D2 D3 D4 NK SY EB CN EM SB EC FS GS RS US
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
SP ?? ?? ?? ?? ?? ?? ?? ?? ?? Ct .  <  (  +  !!
&  ?? ?? ?? ?? ?? ?? ?? ?? ?? !  DO *  )  ;  NO
-  /  ?? ?? ?? ?? ?? ?? ?? ?? BB ,  %  _  >  ?
?? ?? ?? ?? ?? ?? ?? ?? ?? '! :  Nb At '  =  "
?? a  b  c  d  e  f  g  h  i  ?? ?? ?? ?? ?? ??
?? j  k  l  m  n  o  p  q  r  ?? ?? ?? ?? ?? ??
?? '? s  t  u  v  w  x  y  z  ?? ?? ?? ?? ?? ??
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
(! A  B  C  D  E  F  G  H  I  ?? ?? ?? ?? ?? ??
!) J  K  L  M  N  O  P  Q  R  ?? ?? ?? ?? ?? ??
// ?? S  T  U  V  W  X  Y  Z  ?? ?? ?? ?? ?? ??
0  1  2  3  4  5  6  7  8  9  ?? ?? ?? ?? ?? DT
```

5. Character mnemonics

Unambiguous character mnemonics are specified, which provide a practical way of identifying a character, without reference to a coded character set and its code in this coded character set. The mnemonics are written in a minimal set of characters, namely the invariant 83 graphical characters of ISO 646, which is a kind of greatest common subset to be found between the majority of coded character sets, including ASCII, national variants of the ISO 646 7-bit character set and various EBCDICs. In addition, the numeric value of the coded representations of all these characters are the same in all coded character sets compatible with ISO standards. All of them except two, EXCLAMATION MARK and QUOTATION MARK, have the same coded representation in all variants of EBCDIC. This minimal set of characters is called the reference character set in this memo.

5.1. General Syntax

The character mnemonics are taken from the ISO committee draft (CD) of the POSIX.2 standard (3). They are classified into two groups:

1. A group with two-character mnemonics
   Primarily intended for alphabetic scripts like Latin, Greek, Cyrillic, Hebrew and Arabic, and special characters.
2. A group with variable-length mnemonics
   primarily intended for non-alphabetic scripts like Japanese and Chinese, but also used for some accented letters and special characters.

In the two-character mnemonics, all invariant graphic character in the ISO 646 character codes except "&" are used, i.e. the following characters:

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

The character "_" is not used as the first character.

In the variable-length mnemonics, the character "_" is not used as the first character. If it is used in a name, its presence is doubled.

The mnemonics can be used in several different ways for different purposes. One of these is description of coded character sets, which is detailed in section 3. Another is for extending a given coded character set to a mnemonic character set. This is described in section 4. The restrictions on the use of the characters "&" and "_" are due to demands of the compositional methods of these techniques.

5.2. ISO Official Long Descriptive Character Name

For all mnemonics, the character for which it stands is indicated in the following table by a long descriptive name. This name is identical to the ISO name of the character as given in reference (2). For a few characters that are not included there, descriptive names of the same kind are introduced in this memo. The source of each character is stated in the table after the name and should be consulted for a reliable identification of the character.

These long descriptive names consists only of the capital Latin letters of the invariant part of ISO 646, the digits, "-", and SPACE. Digits are only used in names of ideographic and Hangul characters and never as the first character.

5.3. The 2-character Mnemonics

The two-character mnemonics include various accented Latin letters, Greek, Cyrillic, Hebrew, Arabic, Hiragana and Katakana. Also a fair number of special characters are included. Almost all ISO or ISO registered 7- and 8-bit graphical coded character sets are covered with these two-character mnemonics.

The two characters are chosen so the graphical appearence in the reference set resembles as much as possible (within the posibilities available) the graphical appearance of the character. The basic character set of ISO 646 is used as the reference set, as mentioned above.

The characters in the reference character set are chosen to represent themselves.

For control characters from ISO 646 the two-character acronyms of ISO 2047 are used as mnemonics. For the other control characters of ISO 6429, two-character mnemonics have been selected based on the variable-length acronyms used in that standard.

Letters, including Greek, Cyrillic, Arabic and Hebrew, are represented with the base letter as the first letter, and the second letter represents an accent or relation to a non-Latin script. Non-Latin letters are transliterated to Latin letters, following transliteration standards as closely as possible. This is also done with the Latin letters such as ETH and THORN, and the Danish/Norwegian/Swedish letter A WITH RING ABOVE is translitterated into "aa".

After a letter, the second character signifies the following:

Exclamation mark	!	Grave
Apostrophe	’	Acute accent
Greater-Than sign	>	Circumflex accent
Question Mark	?	tilde
Hyphen-Minus	-	Macron
Left parenthesis	(	Breve
Full Stop	.	Dot Above
Colon	:	Diaeresis
Comma	,	Cedilla
Underline	_	Underline
Solidus	/	Stroke
Quotation mark	"	Double acute accent
Semicolon	;	Ogonek
Less-Than sign	<	Caron
Zero	0	Ring above
Two	2	Hook
Nine	9	Horn
Equals	=	Cyrillic
Asterisk	*	Greek
Percent sign	%	Greek/Cyrillic special
Plus	+	smalls: Arabic, capitals: Hebrew
Three	3	some Latin/Greek/Cyrillic letters
Four	4	Bopomofo
Five	5	Hiragana
Six	6	Katakana

In designing the mnemonics the following special characters were reserved: The ampersand is reserved as an intro character, indicating that the following string is in the mnemonic character set. The underscore is reserved for the variable-length mnemonics. This use does not eliminate usage as an accent or language identifier.

Special characters are encoded with some mnemonic value. These are not systematic thruout, but most mnemonics start with a related special character of the reference set.

5.4. The Variable-length Character Mnemonics

The Variable-length Character Mnemonics are primarily meant for the ideographic characters in larger Asian character sets, but are also used for accented characters with several accents and some special characters. To have the mnemonics as short as possible, which both saves storage and is easier to input, a quite short name is preferred. Considering the Chinese standard GB 2312-1980, the Japanese standards JIS X0208 and JIS X0212, and the Korean standard KS C 5601, they are all given by row and column numbers between 1 and 94. So two positions for row and column and a character set identifier of one character would be almost as short as possible. The following character set identifiers are defined:

c GB 2312-1980
j JIS X0208-1990
J JIS X0212-1990
k KS C 5601-1987

This system for the representation of ideographic characters and Hangul characters is not truely mnemonic, but it provides short representations that are easy to connect to the corresponding character by means of the code table of an official character set standard. Alternative methods based on the graphic appearance or the pronunciation of the characters are thought to be unfeasible.

One prominent character in the reference character set is reserved for identifying variable-length mnemonics, namely the underscore "_". This character is intended as a delimiter both in the front and in the end of the mnemonic. An example of its use would be: (&=intro):

&_j3210_ &_j4436_&_j6530_

5.5. Character mnemonic table

The following table contains the character mnemonic and the encoding and long descriptive name of ISO 2DIS 10646 (2). Although the ISO 10646 is only at DIS stage at this moment of writing and there is quite some debate about it, the long descriptive naming in the DIS is considered to be stable and the best official ISO reference to character names. The 2-octet encoded value of the ISO 2DIS 10646 is also used, but only as an identification of the character, and it should only be used for identification purposes as the coded representation may be changed in the final 10646 international standard. Some characters not in the ISO 2DIS 10646 are allocated values in the private use zone and given names and references to a character set where it is used.

The format of the table is:

1st field is the character mnemonic (mostly 2 characters).
2nd field is the ISO 2DIS 10646 code in hexadecimal.
3rd field is the long descriptive name of ISO 2DIS 10646.

NU	0000	NULL (NUL)
SH	0001	START OF HEADING (SOH)
SX	0002	START OF TEXT (STX)
EX	0003	END OF TEXT (ETX)
ET	0004	END OF TRANSMISSION (EOT)
EQ	0005	ENQUIRY (ENQ)
AK	0006	ACKNOWLEDGE (ACK)
BL	0007	BELL (BEL)
BS	0008	BACKSPACE (BS)
HT	0009	CHARACTER TABULATION (HT)
LF	000a	LINE FEED (LF)
VT	000b	LINE TABULATION (VT)
FF	000c	FORM FEED (FF)
CR	000d	CARRIAGE RETURN (CR)
SO	000e	SHIFT OUT (SO)
SI	000f	SHIFT IN (SI)
DL	0010	DATALINK ESCAPE (DLE)
D1	0011	DEVICE CONTROL ONE (DC1)
D2	0012	DEVICE CONTROL TWO (DC2)
D3	0013	DEVICE CONTROL THREE (DC3)
D4	0014	DEVICE CONTROL FOUR (DC4)
NK	0015	NEGATIVE ACKNOWLEDGE (NAK)
SY	0016	SYNCRONOUS IDLE (SYN)
EB	0017	END OF TRANSMISSION BLOCK (ETB)
CN	0018	CANCEL (CAN)
EM	0019	END OF MEDIUM (EM)
SB	001a	SUBSTITUTE (SUB)
EC	001b	ESCAPE (ESC)
FS	001c	FILE SEPARATOR (IS4)
GS	001d	GROUP SEPARATOR (IS3)
RS	001e	RECORD SEPARATOR (IS2)
US	001f	UNIT SEPARATOR (IS1)
SP	0020	SPACE
!	0021	EXCLAMATION MARK
"	0022	QUOTATION MARK
Nb	0023	NUMBER SIGN
DO	0024	DOLLAR SIGN
%	0025	PERCENT SIGN
&	0026	AMPERSAND
'	0027	APOSTROPHE
(	0028	LEFT PARENTHESIS
)	0029	RIGHT PARENTHESIS
*	002a	ASTERISK
+	002b	PLUS SIGN
,	002c	COMMA
-	002d	HYPHEN-MINUS
.	002e	FULL STOP
//	002f	SOLIDUS
0	0030	DIGIT ZERO
1	0031	DIGIT ONE
2	0032	DIGIT TWO
3	0033	DIGIT THREE
4	0034	DIGIT FOUR
5	0035	DIGIT FIVE
6	0036	DIGIT SIX
7	0037	DIGIT SEVEN
8	0038	DIGIT EIGHT
9	0039	DIGIT NINE
:	003a	COLON
;	003b	SEMICOLON
<	003c	LESS-THAN SIGN
=	003d	EQUALS SIGN
>>	003e	GREATER-THAN SIGN
?	003f	QUESTION MARK
At	0040	COMMERCIAL AT
A	0041	LATIN CAPITAL LETTER A
B	0042	LATIN CAPITAL LETTER B
C	0043	LATIN CAPITAL LETTER C
D	0044	LATIN CAPITAL LETTER D
E	0045	LATIN CAPITAL LETTER E
F	0046	LATIN CAPITAL LETTER F
G	0047	LATIN CAPITAL LETTER G
H	0048	LATIN CAPITAL LETTER H
I	0049	LATIN CAPITAL LETTER I
J	004a	LATIN CAPITAL LETTER J
K	004b	LATIN CAPITAL LETTER K
L	004c	LATIN CAPITAL LETTER L
M	004d	LATIN CAPITAL LETTER M
N	004e	LATIN CAPITAL LETTER N
O	004f	LATIN CAPITAL LETTER O
P	0050	LATIN CAPITAL LETTER P
Q	0051	LATIN CAPITAL LETTER Q
R	0052	LATIN CAPITAL LETTER R
S	0053	LATIN CAPITAL LETTER S
T	0054	LATIN CAPITAL LETTER T
U	0055	LATIN CAPITAL LETTER U
V	0056	LATIN CAPITAL LETTER V
W	0057	LATIN CAPITAL LETTER W

X	0058	LATIN CAPITAL LETTER X
Y	0059	LATIN CAPITAL LETTER Y
Z	005a	LATIN CAPITAL LETTER Z
<(	005b	LEFT SQUARE BRACKET
_	005f	LOW LINE
’!	0060	GRAVE ACCENT
a	0061	LATIN SMALL LETTER A
b	0062	LATIN SMALL LETTER B
c	0063	LATIN SMALL LETTER C
d	0064	LATIN SMALL LETTER D
e	0065	LATIN SMALL LETTER E
f	0066	LATIN SMALL LETTER F
g	0067	LATIN SMALL LETTER G
h	0068	LATIN SMALL LETTER H
i	0069	LATIN SMALL LETTER I
j	006a	LATIN SMALL LETTER J
k	006b	LATIN SMALL LETTER K
l	006c	LATIN SMALL LETTER L
m	006d	LATIN SMALL LETTER M
n	006e	LATIN SMALL LETTER N
o	006f	LATIN SMALL LETTER O
p	0070	LATIN SMALL LETTER P
q	0071	LATIN SMALL LETTER Q
r	0072	LATIN SMALL LETTER R
s	0073	LATIN SMALL LETTER S
t	0074	LATIN SMALL LETTER T
u	0075	LATIN SMALL LETTER U
v	0076	LATIN SMALL LETTER V
w	0077	LATIN SMALL LETTER W
x	0078	LATIN SMALL LETTER X
y	0079	LATIN SMALL LETTER Y
z	007a	LATIN SMALL LETTER Z
(!	007b	LEFT CURLY BRACKET
!!	007c	VERTICAL LINE
!)	007d	RIGHT CURLY BRACKET
’?	007e	TILDE
DT	007f	DELETE (DEL)
PA	0080	PADDING CHARACTER (PAD)
HO	0081	HIGH OCTET PRESET (HOP)
BH	0082	BREAK PERMITTED HERE (BPH)
NH	0083	NO BREAK HERE (NBH)
IN	0084	INDEX (IND)
NL	0085	NEXT LINE (NEL)
SA	0086	START OF SELECTED AREA (SSA)
ES	0087	END OF SELECTED AREA (ESA)
HS	0088	CHARACTER TABULATION SET (HTS)
HJ	0089	CHARACTER TABULATION WITH JUSTIFICATION (HTJ)
VS	008a	LINE TABULATION SET (VTS)
PD	008b	PARTIAL LINE FORWARD (PLD)
PU	008c	PARTIAL LINE BACKWARD (PLU)
RI	008d	REVERSE LINE FEED (RI)
S2	008e	SINGLE-SHIFT TWO (SS2)
S3	008f	SINGLE-SHIFT THREE (SS3)
DC	0090	DEVICE CONTROL STRING (DCS)
P1	0091	PRIVATE USE ONE (PU1)
P2	0092	PRIVATE USE TWO (PU2)
TS	0093	SET TRANSMIT STATE (STS)
CC	0094	CANCEL CHARACTER (CCH)
MW	0095	MESSAGE WAITING (MW)
SG	0096	START OF GUARDED AREA (SPA)
EG	0097	END OF GUARDED AREA (EPA)

SS	0098	START OF STRING (SOS)
GC	0099	SINGLE GRAPHIC CHARACTER INTRODUCER (SGCI)
SC	009a	SINGLE CHARACTER INTRODUCER (SCI)
CI	009b	CONTROL SEQUENCE INTRODUCER (CSI)
ST	009c	STRING TERMINATOR (ST)
OC	009d	OPERATING SYSTEM COMMAND (OSC)
PM	009e	PRIVACY MESSAGE (PM)
AC	009f	APPLICATION PROGRAM COMMAND (APC)
NS	00a0	NO-BREAK SPACE
!I	00a1	INVERTED EXCLAMATION MARK
Ct	00a2	CENT SIGN
Pd	00a3	POUND SIGN
Cu	00a4	CURRENCY SIGN
Ye	00a5	YEN SIGN
BB	00a6	BROKEN BAR
SE	00a7	SECTION SIGN
’:	00a8	DIAERESIS
Co	00a9	COPYRIGHT SIGN
-a	00aa	FEMININE ORDINAL INDICATOR
<<	00ab	LEFT-POINTING DOUBLE ANGLE QUOTATION MARK
NO	00ac	NOT SIGN
--	00ad	SOFT HYPHEN
Rg	00ae	REGISTERED SIGN
DG	00b0	DEGREE SIGN
+-	00b1	PLUS-MINUS SIGN
2S	00b2	SUPERSCRIPT TWO
3S	00b3	SUPERSCRIPT THREE
’’	00b4	ACUTE ACCENT
My	00b5	MICRO SIGN
PI	00b6	PILCROW SIGN
.M	00b7	MIDDLE DOT
’,	00b8	CEDILLA
1S	00b9	SUPERSCRIPT ONE
-o	00ba	MASCULINE ORDINAL INDICATOR
14	00bc	VULGAR FRACTION ONE QUARTER
12	00bd	VULGAR FRACTION ONE HALF
34	00be	VULGAR FRACTION THREE QUARTERS
?I	00bf	INVERTED QUESTION MARK
A!	00c0	LATIN CAPITAL LETTER A WITH GRAVE
A’	00c1	LATIN CAPITAL LETTER A WITH ACUTE
A?	00c3	LATIN CAPITAL LETTER A WITH TILDE
A:	00c4	LATIN CAPITAL LETTER A WITH DIAERESIS
AA	00c5	LATIN CAPITAL LETTER A WITH RING ABOVE
AE	00c6	LATIN CAPITAL LETTER AE
C,	00c7	LATIN CAPITAL LETTER C WITH CEDILLA
E!	00c8	LATIN CAPITAL LETTER E WITH GRAVE
E’	00c9	LATIN CAPITAL LETTER E WITH ACUTE
E:	00cb	LATIN CAPITAL LETTER E WITH DIAERESIS

I!	00cc	LATIN CAPITAL LETTER I WITH GRAVE
I’	00cd	LATIN CAPITAL LETTER I WITH ACUTE
I:	00cf	LATIN CAPITAL LETTER I WITH DIAERESIS
D-	00d0	LATIN CAPITAL LETTER ETH (Icelandic)
N?	00d1	LATIN CAPITAL LETTER N WITH TILDE
O!	00d2	LATIN CAPITAL LETTER O WITH GRAVE
O’	00d3	LATIN CAPITAL LETTER O WITH ACUTE
O?	00d5	LATIN CAPITAL LETTER O WITH TILDE
O:	00d6	LATIN CAPITAL LETTER O WITH DIAERESIS
*X	00d7	MULTIPLICATION SIGN
U!	00d9	LATIN CAPITAL LETTER U WITH GRAVE
U’	00da	LATIN CAPITAL LETTER U WITH ACUTE
U:	00dc	LATIN CAPITAL LETTER U WITH DIAERESIS
Y’	00dd	LATIN CAPITAL LETTER Y WITH ACUTE
TH	00de	LATIN CAPITAL LETTER THORN (Icelandic)
ss	00df	LATIN SMALL LETTER SHARP S (German)
a!	00e0	LATIN SMALL LETTER A WITH GRAVE
a’	00e1	LATIN SMALL LETTER A WITH ACUTE
a?	00e3	LATIN SMALL LETTER A WITH TILDE
a:	00e4	LATIN SMALL LETTER A WITH DIAERESIS
aa	00e5	LATIN SMALL LETTER A WITH RING ABOVE
ae	00e6	LATIN SMALL LETTER AE
c,	00e7	LATIN SMALL LETTER C WITH CEDILLA
e!	00e8	LATIN SMALL LETTER E WITH GRAVE
e’	00e9	LATIN SMALL LETTER E WITH ACUTE
e:	00eb	LATIN SMALL LETTER E WITH DIAERESIS
i!	00ec	LATIN SMALL LETTER I WITH GRAVE
i’	00ed	LATIN SMALL LETTER I WITH ACUTE
i:	00ef	LATIN SMALL LETTER I WITH DIAERESIS
d-	00f0	LATIN SMALL LETTER ETH (Icelandic)
n?	00f1	LATIN SMALL LETTER N WITH TILDE
o!	00f2	LATIN SMALL LETTER O WITH GRAVE
o’	00f3	LATIN SMALL LETTER O WITH ACUTE
o?	00f5	LATIN SMALL LETTER O WITH TILDE
o:	00f6	LATIN SMALL LETTER O WITH DIAERESIS
-:	00f7	DIVISION SIGN
u!	00f9	LATIN SMALL LETTER U WITH GRAVE
u’	00fa	LATIN SMALL LETTER U WITH ACUTE
u:	00fc	LATIN SMALL LETTER U WITH DIAERESIS
y’	00fd	LATIN SMALL LETTER Y WITH ACUTE
th	00fe	LATIN SMALL LETTER THORN (Icelandic)
y:	00ff	LATIN SMALL LETTER Y WITH DIAERESIS
A-	0100	LATIN CAPITAL LETTER A WITH MACRON
a-	0101	LATIN SMALL LETTER A WITH MACRON
A(	0102	LATIN CAPITAL LETTER A WITH BREVE
a(	0103	LATIN SMALL LETTER A WITH BREVE
A;	0104	LATIN CAPITAL LETTER A WITH OGONEK
a;	0105	LATIN SMALL LETTER A WITH OGONEK
C’	0106	LATIN CAPITAL LETTER C WITH ACUTE
c’	0107	LATIN SMALL LETTER C WITH ACUTE
C.	010a	LATIN CAPITAL LETTER C WITH DOT ABOVE
c.	010b	LATIN SMALL LETTER C WITH DOT ABOVE
C<	010c	LATIN CAPITAL LETTER C WITH CARON
c<	010d	LATIN SMALL LETTER C WITH CARON
D<	010e	LATIN CAPITAL LETTER D WITH CARON
d<	010f	LATIN SMALL LETTER D WITH CARON
E-	0112	LATIN CAPITAL LETTER E WITH MACRON
e-	0113	LATIN SMALL LETTER E WITH MACRON
E(	0114	LATIN CAPITAL LETTER E WITH BREVE
e(	0115	LATIN SMALL LETTER E WITH BREVE
E.	0116	LATIN CAPITAL LETTER E WITH DOT ABOVE
e.	0117	LATIN SMALL LETTER E WITH DOT ABOVE
E;	0118	LATIN CAPITAL LETTER E WITH OGONEK
e;	0119	LATIN SMALL LETTER E WITH OGONEK
E<	011a	LATIN CAPITAL LETTER E WITH CARON
e<	011b	LATIN SMALL LETTER E WITH CARON
G(	011e	LATIN CAPITAL LETTER G WITH BREVE
g(	011f	LATIN SMALL LETTER G WITH BREVE
G.	0120	LATIN CAPITAL LETTER G WITH DOT ABOVE
g.	0121	LATIN SMALL LETTER G WITH DOT ABOVE
G,	0122	LATIN CAPITAL LETTER G WITH CEDILLA
g,	0123	LATIN SMALL LETTER G WITH CEDILLA
I?	0128	LATIN CAPITAL LETTER I WITH TILDE
i?	0129	LATIN SMALL LETTER I WITH TILDE
I-	012a	LATIN CAPITAL LETTER I WITH MACRON
i-	012b	LATIN SMALL LETTER I WITH MACRON
I;	012e	LATIN CAPITAL LETTER I WITH OGONEK

i;	012f	LATIN SMALL LETTER I WITH OGONEK
I.	0130	LATIN CAPITAL LETTER I WITH DOT ABOVE
i.	0131	LATIN SMALL LETTER I WITH NO DOT
IJ	0132	LATIN CAPITAL LIGATURE IJ
ij	0133	LATIN SMALL LIGATURE IJ
K,	0136	LATIN CAPITAL LETTER K WITH CEDILLA
k,	0137	LATIN SMALL LETTER K WITH CEDILLA
kk	0138	LATIN SMALL LETTER KRA (Greenlandic)
L’	0139	LATIN CAPITAL LETTER L WITH ACUTE
l’	013a	LATIN SMALL LETTER L WITH ACUTE
L,	013b	LATIN CAPITAL LETTER L WITH CEDILLA
l,	013c	LATIN SMALL LETTER L WITH CEDILLA
L<	013d	LATIN CAPITAL LETTER L WITH CARON
l<	013e	LATIN SMALL LETTER L WITH CARON
L.	013f	LATIN CAPITAL LETTER L WITH MIDDLE DOT
l.	0140	LATIN SMALL LETTER L WITH MIDDLE DOT
N’	0143	LATIN CAPITAL LETTER N WITH ACUTE
n’	0144	LATIN SMALL LETTER N WITH ACUTE
N,	0145	LATIN CAPITAL LETTER N WITH CEDILLA
n,	0146	LATIN SMALL LETTER N WITH CEDILLA
N<	0147	LATIN CAPITAL LETTER N WITH CARON
n<	0148	LATIN SMALL LETTER N WITH CARON
’n	0149	LATIN SMALL LETTER N PRECEDED BY APOSTROPHE
NG	014a	LATIN CAPITAL LETTER ENG (Lappish)
ng	014b	LATIN SMALL LETTER ENG
O-	014c	LATIN CAPITAL LETTER O WITH MACRON
o-	014d	LATIN SMALL LETTER O WITH MACRON
O(	014e	LATIN CAPITAL LETTER O WITH BREVE
o(	014f	LATIN SMALL LETTER O WITH BREVE
O"	0150	LATIN CAPITAL LETTER O WITH DOUBLE ACUTE
o"	0151	LATIN SMALL LETTER O WITH DOUBLE ACUTE
OE	0152	LATIN CAPITAL LIGATURE OE
oe	0153	LATIN SMALL LIGATURE OE
R’	0154	LATIN CAPITAL LETTER R WITH ACUTE
r’	0155	LATIN SMALL LETTER R WITH ACUTE
R,	0156	LATIN CAPITAL LETTER R WITH CEDILLA
r,	0157	LATIN SMALL LETTER R WITH CEDILLA
R<	0158	LATIN CAPITAL LETTER R WITH CARON
r<	0159	LATIN SMALL LETTER R WITH CARON
S’	015a	LATIN CAPITAL LETTER S WITH ACUTE
s’	015b	LATIN SMALL LETTER S WITH ACUTE
S,	015e	LATIN CAPITAL LETTER S WITH CEDILLA
s,	015f	LATIN SMALL LETTER S WITH CEDILLA
S<	0160	LATIN CAPITAL LETTER S WITH CARON
s<	0161	LATIN SMALL LETTER S WITH CARON
T,	0162	LATIN CAPITAL LETTER T WITH CEDILLA
t,	0163	LATIN SMALL LETTER T WITH CEDILLA
T<	0164	LATIN CAPITAL LETTER T WITH CARON
t<	0165	LATIN SMALL LETTER T WITH CARON
U?	0168	LATIN CAPITAL LETTER U WITH TILDE
u?	0169	LATIN SMALL LETTER U WITH TILDE
U-	016a	LATIN CAPITAL LETTER U WITH MACRON
u-	016b	LATIN SMALL LETTER U WITH MACRON
U(	016c	LATIN CAPITAL LETTER U WITH BREVE
u(	016d	LATIN SMALL LETTER U WITH BREVE
U0	016e	LATIN CAPITAL LETTER U WITH RING ABOVE
u0	016f	LATIN SMALL LETTER U WITH RING ABOVE
U"	0170	LATIN CAPITAL LETTER U WITH DOUBLE ACUTE
u"	0171	LATIN SMALL LETTER U WITH DOUBLE ACUTE
U;	0172	LATIN CAPITAL LETTER U WITH OGONEK
u;	0173	LATIN SMALL LETTER U WITH OGONEK
Y:	0178	LATIN CAPITAL LETTER Y WITH DIAERESIS
Z’	0179	LATIN CAPITAL LETTER Z WITH ACUTE
z’	017a	LATIN SMALL LETTER Z WITH ACUTE
Z.	017b	LATIN CAPITAL LETTER Z WITH DOT ABOVE
z.	017c	LATIN SMALL LETTER Z WITH DOT ABOVE
Z<	017d	LATIN CAPITAL LETTER Z WITH CARON
z<	017e	LATIN SMALL LETTER Z WITH CARON
ED	01b7	LATIN CAPITAL LETTER EZH
A1	01de	LATIN CAPITAL LETTER A WITH MACRON AND DIAERESIS
a1	01df	LATIN SMALL LETTER A WITH MACRON AND DIAERESIS
A7	01e0	LATIN CAPITAL LETTER A WITH MACRON AND DOT ABOVE
a7	01e1	LATIN SMALL LETTER A WITH MACRON AND DOT ABOVE
A3	01e2	LATIN CAPITAL LETTER AE WITH MACRON
a3	01e3	LATIN SMALL LETTER AE WITH

		MACRON
G<	01e6	LATIN CAPITAL LETTER G WITH CARON
g<	01e7	LATIN SMALL LETTER G WITH CARON
K<	01e8	LATIN CAPITAL LETTER K WITH CARON
k<	01e9	LATIN SMALL LETTER K WITH CARON
O;	01ea	LATIN CAPITAL LETTER O WITH OGONEK
o;	01eb	LATIN SMALL LETTER O WITH OGONEK
O1	01ec	LATIN CAPITAL LETTER O WITH MACRON AND OGONEK
o1	01ed	LATIN SMALL LETTER O WITH MACRON AND OGONEK
EZ	01ee	LATIN CAPITAL LETTER EZH WITH CARON
ez	01ef	LATIN SMALL LETTER EZH WITH CARON
AA'	01fa	LATIN CAPITAL LETTER A WITH ACUTE AND RING ABOVE
aa'	01fb	LATIN SMALL LETTER A WITH ACUTE AND RING ABOVE
ae'	01fd	LATIN SMALL LETTER AE WITH ACUTE
;S	02bf	MODIFIER LETTER LEFT HALF RING
'<	02c7	CARON
'(	02d8	BREVE
'.	02d9	DOT ABOVE
'0	02da	RING ABOVE
';	02db	OGONEK
'"	02dd	DOUBLE ACUTE ACCENT
G*	0393	GREEK CAPITAL LETTER GAMMA
D*	0394	GREEK CAPITAL LETTER DELTA
H*	0398	GREEK CAPITAL LETTER THETA
S*	03a3	GREEK CAPITAL LETTER SIGMA
F*	03a6	GREEK CAPITAL LETTER PHI
W*	03a9	GREEK CAPITAL LETTER OMEGA
a*	03b1	GREEK SMALL LETTER ALPHA
b*	03b2	GREEK SMALL LETTER BETA
d*	03b4	GREEK SMALL LETTER DELTA
e*	03b5	GREEK SMALL LETTER EPSILON
m*	03bc	GREEK SMALL LETTER MU
p*	03c0	GREEK SMALL LETTER PI
s*	03c3	GREEK SMALL LETTER SIGMA
t*	03c4	GREEK SMALL LETTER TAU
-N	2013	EN DASH
-M	2014	EM DASH
'6	2018	LEFT SINGLE QUOTATION MARK
'9	2019	RIGHT SINGLE QUOTATION MARK
.9	201a	SINGLE LOW-9 QUOTATION MARK
"6	201c	LEFT DOUBLE QUOTATION MARK
"9	201d	RIGHT DOUBLE QUOTATION MARK
:9	201e	DOUBLE LOW-9 QUOTATION MARK
%0	2030	PER MILLE SIGN
1"	2035	REVERSED PRIME
<1	2039	SINGLE LEFT-POINTING ANGLE QUOTATION MARK
'-	203e	OVERLINE
nS	207f	SUPERSCRIPT LATIN SMALL LETTER N
Li	20a4	ITALIAN LIRA SIGN
Pt	20a7	PESETA SYMBOL
TM	2122	TRADE MARK SIGN
Om	2126	OHM SIGN
18	215b	VULGAR FRACTION ONE EIGHTH
38	215c	VULGAR FRACTION THREE EIGHTHS
58	215d	VULGAR FRACTION FIVE EIGHTHS
78	215e	VULGAR FRACTION SEVEN EIGHTHS
<-	2190	LEFTWARDS ARROW
-!	2191	UPWARDS ARROW
-v	2193	DOWNWARDS ARROW
==	21d4	LEFT RIGHT DOUBLE ARROW
dP	2202	PARTIAL DIFFERENTIAL
*P	220f	N-ARY PRODUCT
+Z	2211	N-ARY SUMMATION
Ob	2218	RING OPERATOR
Sb	2219	BULLET OPERATOR
RT	221a	SQUARE ROOT
00	221e	INFINITY
AN	2227	LOGICAL AND
(U	2229	INTERSECTION
In	222b	INTEGRAL
?1	223c	TILDE OPERATOR
?2	2248	ALMOST EQUAL TO
!=	2260	NOT EQUAL TO
=3	2261	IDENTICAL TO
=<	2264	LESS-THAN OR EQUAL TO
.3	22ef	MIDLINE HORIZONTAL ELLIPSIS
NI	2310	REVERSED NOT SIGN
Iu	2320	TOP HALF INTEGRAL
Il	2321	BOTTOM HALF INTEGRAL
hh	2500	BOX DRAWINGS LIGHT HORIZONTAL
HH	2501	BOX DRAWINGS HEAVY HORIZONTAL
vv	2502	BOX DRAWINGS LIGHT VERTICAL
VV	2503	BOX DRAWINGS HEAVY VERTICAL
dr	250c	BOX DRAWINGS LIGHT DOWN AND RIGHT
dR	250d	BOX DRAWINGS DOWN LIGHT AND RIGHT HEAVY
Dr	250e	BOX DRAWINGS DOWN HEAVY AND RIGHT LIGHT
DR	250f	BOX DRAWINGS HEAVY DOWN AND RIGHT
dl	2510	BOX DRAWINGS LIGHT DOWN AND LEFT
dL	2511	BOX DRAWINGS DOWN LIGHT AND LEFT HEAVY
Dl	2512	BOX DRAWINGS DOWN HEAVY AND LEFT LIGHT
LD	2513	BOX DRAWINGS HEAVY DOWN AND LEFT
ur	2514	BOX DRAWINGS LIGHT UP AND RIGHT
uR	2515	BOX DRAWINGS UP LIGHT AND RIGHT HEAVY
Ur	2516	BOX DRAWINGS UP HEAVY AND RIGHT LIGHT
UR	2517	BOX DRAWINGS HEAVY UP AND RIGHT
ul	2518	BOX DRAWINGS LIGHT UP AND LEFT
uL	2519	BOX DRAWINGS UP LIGHT AND LEFT HEAVY
Ul	251a	BOX DRAWINGS UP HEAVY AND LEFT LIGHT
UL	251b	BOX DRAWINGS HEAVY UP AND LEFT
vr	251c	BOX DRAWINGS LIGHT VERTICAL AND RIGHT
vR	251d	BOX DRAWINGS VERTICAL LIGHT AND RIGHT HEAVY
Vr	2520	BOX DRAWINGS VERTICAL HEAVY AND

		RIGHT LIGHT
VR	2523	BOX DRAWINGS HEAVY VERTICAL AND RIGHT
vl	2524	BOX DRAWINGS LIGHT VERTICAL AND LEFT
vL	2527	BOX DRAWINGS VERTICAL LIGHT AND LEFT HEAVY
Vl	2528	BOX DRAWINGS VERTICAL HEAVY AND LEFT LIGHT
VL	252b	BOX DRAWINGS HEAVY VERTICAL AND LEFT
dh	252c	BOX DRAWINGS LIGHT DOWN AND HORIZONTAL
dH	252f	BOX DRAWINGS DOWN LIGHT AND HORIZONTAL HEAVY
Dh	2530	BOX DRAWINGS DOWN HEAVY AND HORIZONTAL LIGHT
DH	2533	BOX DRAWINGS HEAVY DOWN AND HORIZONTAL
uh	2534	BOX DRAWINGS LIGHT HORIZONTAL AND UP
uH	2537	BOX DRAWINGS HORIZONTAL HEAVY AND UP LIGHT
Uh	2538	BOX DRAWINGS HORIZONTAL LIGHT AND UP HEAVY
UH	253b	BOX DRAWINGS HEAVY HORIZONTAL AND UP
vh	253c	BOX DRAWINGS LIGHT VERTICAL AND HORIZONTAL
vH	253f	BOX DRAWINGS VERTICAL LIGHT AND HORIZONTAL HEAVY
Vh	2542	BOX DRAWINGS VERTICAL HEAVY AND HORIZONTAL LIGHT
VH	254b	BOX DRAWINGS HEAVY VERTICAL AND HORIZONTAL
TB	2580	UPPER HALF BLOCK
LB	2584	LOWER HALF BLOCK
FB	2588	FULL BLOCK
lB	258c	LEFT HALF BLOCK
RB	2590	RIGHT HALF BLOCK
.S	2591	LIGHT SHADE
:S	2592	MEDIUM SHADE
?S	2593	DARK SHADE
fS	25a0	BLACK SQUARE
Db	25c6	BLACK DIAMOND
Md	2669	QUARTER NOTE
"!	e006	NON-SPACING GRAVE ACCENT (ISO-IR-103 193)
"'	e007	NON-SPACING ACUTE ACCENT (ISO-IR-103 194)
"?	e009	NON-SPACING TILDE (ISO-IR-103 196)
"-	e00a	NON-SPACING MACRON (ISO-IR-103 197)
"(	e00b	NON-SPACING BREVE (ISO-IR-103 198)
".	e00c	NON-SPACING DOT ABOVE (ISO-IR-103 199)
":	e00d	NON-SPACING DIAERESIS (ISO-IR-103 200)
"0	e00e	NON-SPACING RING ABOVE (ISO-IR-103 202)
""	e00f	NON-SPACING DOUBLE ACCUTE ACCENT (ISO-IR-103 204)
"<	e010	NON-SPACING CARON (ISO-IR-103 206)
",	e011	NON-SPACING CEDILLA (ISO-IR-103 203)
";	e012	NON-SPACING OGONEK (ISO-IR-103 206)
"_	e013	NON-SPACING LOW LINE (ISO-IR-103 204)
ed	e01e	LATIN SMALL LETTER EZH (ISO-IR-158 142)
Fl	e023	DUTCH GUILDER SIGN (IBM437 159)
fi	fb01	LATIN SMALL LIGATURE FI
fl	fb02	LATIN SMALL LIGATURE FL

6. English language locale for Denmark

```
escape_char/
comment_char    %

% English language locale for Denmark
%
% Written according to POSIX.2
% ISO/IEC 9945-2 2CD specifications
%
% Source: Danish Standards Association
% Revision 1.3 1992-04-19


LC_CTYPE

digit   <0>;<1>;<2>;<3>;<4>;/
        <5>;<6>;<7>;<8>;<9>

xdigit  <0>;<1>;<2>;<3>;<4>;/
        <5>;<6>;<7>;<8>;<9>;/
        <A>;<B>;<C>;<D>;<E>;<F>;/
        <a>;<b>;<c>;<d>;<e>;<f>

blank   <SP>;<HT>;<NS>

space   <SP>;<LF>;<VT>;<FF>;/
        <CR>;<HT>;<NS>

upper   <A>;<B>;<C>;<D>;<E>;<F>;/
        <G>;<H>;<I>;<J>;<K>;<L>;/
        <M>;<N>;<O>;<P>;<Q>;<R>;/
        <S>;<T>;<U>;<V>;<W>;<X>;/
        <Y>;<Z>;<A!>;<A'>;<A?>;<A:>;/
        <AA>;<AE>;<C,>;<E!>;<E'>;<E:>;/
        <I!>;<I'>;<I:>;<D->;<N?>;<O!>;/
        <O'>;<O?>;<O:>;<U!>;<U'>;<U:>;/
        <Y'>;<TH>;<A->;<A(>;<A;>;<C'>;/
        <C<>;<E->;<E(>;<E.>;<E;>;<G,>;/
        <I?>;<I->;<I;>;<IJ>;<K,>;<L'>;/
        <L,>;<L.>;<N'>;<N,>;<NG>;<O->;/
        <O(>;<OE>;<R'>;<R,>;<R<>;<S'>;/
        <S,>;<S<>;<T,>;<T<>;<U?>;<U->;/
        <U;>;<Y:>;<Z'>;<Z<>;<ED>;<A1>;/
        <A7>;<A3>;<G<>;<K<>;<O;>;<O1>;/
        <EZ>;<AA'>;<AE'>;<G*>;<D*>;<H*>;/
        <S*>;<F*>;<W*>

lower   <a>;<b>;<c>;<d>;<e>;<f>;/
        <g>;<h>;<i>;<j>;<k>;<l>;/
        <m>;<n>;<o>;<p>;<q>;<r>;/
        <s>;<t>;<u>;<v>;<w>;<x>;/
        <y>;<z>;<ss>;<a!>;<a'>;<a?>;/
        <a:>;<aa>;<ae>;<c,>;<e!>;<e'>;/
        <e:>;<i!>;<i'>;<i:>;<d->;<n?>;/
        <o!>;<o'>;<o?>;<o:>;<u!>;<u'>;/
        <u:>;<y'>;<th>;<y:>;<a->;<a(>;/
        <a;>;<c'>;<c<>;<e->;<e(>;<e.>;/
        <e;>;<g,>;<i?>;<i->;<i;>;<i.>;/
        <ij>;<k,>;<kk>;<l'>;<l,>;<l.>;/
        <n'>;<n,>;<'n>;<ng>;<o->;<o(>;/
        <oe>;<r'>;<r,>;<r<>;<s'>;<s,>;/
        <s<>;<t,>;<t<>;<u?>;<u->;<u;>;/
        <z'>;<z<>;<a1>;<a7>;<a3>;<g<>;/
        <k<>;<o;>;<o1>;<ez>;<aa'>;<ae'>;/
        <a*>;<b*>;<d*>;<e*>;<m*>;<p*>;/
        <s*>;<t*>;<nS>;<ed>;<a8>;<b8>;/
        <c8>;<d8>;<e8>;<f8>;<g8>;<h8>;/
        <i8>;<j8>;<k8>;<l8>;<m8>;<n8>;/
        <o8>;<p8>;<q8>;<r8>;<s8>;<t8>;/
        <u8>;<v8>;<w8>;<x8>;<y8>;<z8>;/
        <fi>;<fl>

alpha   <A>;<B>;<C>;<D>;<E>;<F>;/
        <G>;<H>;<I>;<J>;<K>;<L>;/
        <M>;<N>;<O>;<P>;<Q>;<R>;/
        <S>;<T>;<U>;<V>;<W>;<X>;/
        <Y>;<Z>;<a>;<b>;<c>;<d>;/
        <e>;<f>;<g>;<h>;<i>;<j>;/
        <k>;<l>;<m>;<n>;<o>;<p>;/
        <q>;<r>;<s>;<t>;<u>;<v>;/
        <w>;<x>;<y>;<z>;<-->;<A!>;/
        <A'>;<A?>;<A:>;<AA>;<AE>;<C,>;/
        <E!>;<E'>;<E:>;<I!>;<I'>;<I:>;/
        <D->;<N?>;<O!>;<O'>;<O?>;<O:>;/
        <U!>;<U'>;<U:>;<Y'>;<TH>;<ss>;/
        <a!>;<a'>;<a?>;<a:>;<aa>;<ae>;/
        <c,>;<e!>;<e'>;<e:>;<i!>;<i'>;/
        <i:>;<d->;<n?>;<o!>;<o'>;<o?>;/
        <o:>;<u!>;<u'>;<u:>;<y'>;<th>;/
        <y:>;<A->;<a->;<A(>;<a(>;<A;>;/
        <a;>;<C'>;<c'>;<C<>;<c<>;<E->;/
        <e->;<E(>;<e(>;<E.>;<e.>;<E;>;/
        <e;>;<G,>;<g,>;<I?>;<i?>;<I->;/
        <i->;<I;>;<i;>;<i.>;<IJ>;<ij>;/
        <K,>;<k,>;<kk>;<L'>;<l'>;<L,>;/
        <l,>;<L.>;<l.>;<N'>;<n'>;<N,>;/
        <n,>;<'n>;<NG>;<ng>;<O->;<o->;/
        <O(>;<o(>;<OE>;<oe>;<R'>;<r'>;/
        <R,>;<r,>;<R<>;<r<>;<S'>;<s'>;/
        <S,>;<s,>;<S<>;<s<>;<T,>;<t,>;/
        <T<>;<t<>;<U?>;<u?>;<U->;<u->;/
        <U;>;<u;>;<Y:>;<Z'>;<z'>;<Z<>;/
        <z<>;<ED>;<A1>;<a1>;<A7>;<a7>;/
        <A3>;<a3>;<G<>;<g<>;<K<>;<k<>;/
        <O;>;<o;>;<O1>;<o1>;<EZ>;<ez>;/
        <AA'>;<aa'>;<AE'>;<ae'>;<;S>;<G*>;/
        <D*>;<H*>;<S*>;<F*>;<W*>;<a*>;/
        <b*>;<d*>;<e*>;<m*>;<p*>;<s*>;/
        <t*>;<nS>;<ed>;<a8>;<b8>;<c8>;/
        <d8>;<e8>;<f8>;<g8>;<h8>;<i8>;/
        <j8>;<k8>;<l8>;<m8>;<n8>;<o8>;/
        <p8>;<q8>;<r8>;<s8>;<t8>;<u8>;/
        <v8>;<w8>;<x8>;<y8>;<z8>;<fi>;/
        <fl>
```

```
cntrl   <NU>;<SH>;<SX>;<EX>;<ET>;<EQ>;/
        <AK>;<BL>;<BS>;<HT>;<LF>;<VT>;/
        <FF>;<CR>;<SO>;<SI>;<DL>;<D1>;/
        <D2>;<D3>;<D4>;<NK>;<SY>;<EB>;/
        <CN>;<EM>;<SB>;<EC>;<FS>;<GS>;/
        <RS>;<US>;<DT>;<PA>;<HO>;<BH>;/
        <NH>;<IN>;<NL>;<SA>;<ES>;<HS>;/
        <HJ>;<VS>;<PD>;<PU>;<RI>;<S2>;/
        <S3>;<DC>;<P1>;<P2>;<TS>;<CC>;/
        <MW>;<SG>;<EG>;<SS>;<GC>;<SC>;/
        <CI>;<ST>;<OC>;<PM>;<AC>

punct   <NU>;<SH>;<SX>;<EX>;<ET>;<EQ>;/
        <AK>;<BL>;<BS>;<HT>;<LF>;<VT>;/
        <FF>;<CR>;<SO>;<SI>;<DL>;<D1>;/
        <D2>;<D3>;<D4>;<NK>;<SY>;<EB>;/
        <CN>;<EM>;<SB>;<EC>;<FS>;<GS>;/
        <RS>;<US>;<!>;<">;<Nb>;<DO>;/
        <%>;<&>;<’>;<(>;<)>;<*>;/
        <+>;<,>;<->;<.>;<//>;<:>;/
        <;>;<<>;<=>;</>>;<?>;<At>;/
        <<(>;<_>;<’!>;<(!>;<!!>;<!)>;/
        <’?>;<DT>;<PA>;<HO>;<BH>;<NH>;/
        <IN>;<NL>;<SA>;<ES>;<HS>;<HJ>;/
        <VS>;<PD>;<PU>;<RI>;<S2>;<S3>;/
        <DC>;<P1>;<P2>;<TS>;<CC>;<MW>;/
        <SG>;<EG>;<SS>;<GC>;<SC>;<CI>;/
        <ST>;<OC>;<PM>;<AC>;<!I>;<Ct>;/
        <Pd>;<Cu>;<Ye>;<BB>;<SE>;<’:>;/
        <Co>;<-a>;<<<>;<NO>;<Rg>;<DG>;/
        <+->;<2S>;<3S>;<’’>;<My>;<PI>;/
        <.M>;<’,>;<1S>;<-o>;<14>;<12>;/
        <34>;<?I>;<*X>;<-:>;<’<>;<’(>;/
        <’.>;<’0>;<’;>;<’">;<-N>;<-M>;/
        <’6>;<’9>;<.9>;<"6>;<"9>;<:9>;/
        <%0>;<1">;<Ca>;<<1>;<’->;<Li>;/
        <Pt>;<TM>;<Om>;<18>;<38>;<58>;/
        <78>;<<->;<-!>;<-v>;<==>;<dP>;/
        <*P>;<+Z>;<Ob>;<Sb>;<RT>;<00>;/
        <AN>;<(U>;<In>;<?1>;<?2>;<!=>;/
        <=3>;<=<>;<.3>;<NI>;<Iu>;<Il>;/
        <hh>;<HH>;<vv>;<VV>;<dr>;<dR>;/
        <Dr>;<DR>;<dl>;<dL>;<Dl>;<LD>;/
        <ur>;<uR>;<Ur>;<UR>;<ul>;<uL>;/
        <Ul>;<UL>;<vr>;<vR>;<Vr>;<VR>;/
        <vl>;<vL>;<Vl>;<VL>;<dh>;<dH>;/
        <Dh>;<DH>;<uh>;<uH>;<Uh>;<UH>;/
        <vh>;<vH>;<Vh>;<VH>;<TB>;<LB>;/
        <FB>;<lB>;<RB>;<.S>;<:S>;<?S>;/
        <fS>;<Db>;<Md>;<"!>;<"’>;<"?>;/
        <"->;<"(>;<".>;<":>;<"0>;<"">;/
        <"<>;<",>;<";>;<"_>;<Fl>

tolower (<A>,<a>); (<A!>,<a!>);/
        (<A’>,<a’>); (<A(>,<a(>);/
        (<A->,<a->); (<A1>,<a1>);/
        (<A3>,<a3>); (<A7>,<a7>);/
        (<A:>,<a:>); (<A;>,<a;>);/
        (<A?>,<a?>); (<AA>,<aa>);/
        (<AA’>,<aa’>); (<AE>,<ae>);/
        (<AE’>,<ae’>); (<B>,<b>);/
        (<C>,<c>); (<C’>,<c’>);/
        (<C,>,<c,>); (<C<>,<c<>);/
        (<D>,<d>); (<E>,<e>);/
        (<E!>,<e!>); (<E’>,<e’>);/
        (<E(>,<e(>); (<E->,<e->);/
        (<E.>,<e.>); (<E:>,<e:>);/
        (<E;>,<e;>); (<ED>,<ed>);/
        (<EZ>,<ez>); (<F>,<f>);/
        (<G>,<g>); (<G,>,<g,>);/
        (<G<>,<g<>); (<H>,<h>);/
        (<J>,<j>); (<K>,<k>);/
        (<K,>,<k,>); (<K<>,<k<>);/
        (<L>,<l>); (<L’>,<l’>);/
        (<L,>,<l,>); (<L.>,<l.>);/
        (<M>,<m>); (<N>,<n>);/
        (<N’>,<n’>); (<N,>,<n,>);/
        (<N?>,<n?>); (<O>,<o>);/
        (<O!>,<o!>); (<O’>,<o’>);/
        (<O(>,<o(>); (<O->,<o->);/
        (<O1>,<o1>); (<O:>,<o:>);/
        (<O;>,<o;>); (<O?>,<o?>);/
        (<OE>,<oe>); (<P>,<p>);/
        (<Q>,<q>); (<R>,<r>);/
        (<R’>,<r’>); (<R,>,<r,>);/
        (<R<>,<r<>); (<S>,<s>);/
        (<S’>,<s’>); (<S,>,<s,>);/
        (<S<>,<s<>); (<T>,<t>);/
        (<T,>,<t,>); (<T<>,<t<>);/
        (<U>,<u>); (<U!>,<u!>);/
        (<U’>,<u’>); (<U->,<u->);/
        (<U:>,<u:>); (<U;>,<u;>);/
        (<U?>,<u?>); (<V>,<v>);/
        (<W>,<w>); (<X>,<x>);/
        (<Y>,<y>); (<Y’>,<y’>);/
        (<Z>,<z>); (<Z’>,<z’>);/
        (<Z<>,<z<>);

toupper (<a>,<A>); (<a!>,<A!>);/
        (<a’>,<A’>); (<a(>,<A(>);/
        (<a->,<A->); (<a1>,<A1>);/
        (<a3>,<A3>); (<a7>,<A7>);/
        (<a:>,<A:>); (<a;>,<A;>);/
        (<a?>,<A?>); (<aa>,<AA>);/
        (<aa’>,<AA’>); (<ae>,<AE>);/
        (<ae’>,<AE’>); (<b>,<B>);/
        (<c>,<C>); (<c’>,<C’>);/
        (<c,>,<C,>); (<c<>,<C<>);/
        (<d>,<D>); (<e>,<E>);/
        (<e!>,<E!>); (<e’>,<E’>);/
        (<e(>,<E(>); (<e->,<E->);/
        (<e.>,<E.>); (<e:>,<E:>);/
        (<e;>,<E;>); (<ed>,<ED>);/
        (<ez>,<EZ>); (<f>,<F>);/
```

```
        (<g>,<G>); (<g,>,<G,>);/
        (<g<>,<G<>); (<h>,<H>);/
        (<j>,<J>); (<k>,<K>);/
        (<k,>,<K,>); (<k<>,<K<>);/
        (<l>,<L>); (<l'>,<L'>);/
        (<l,>,<L,>); (<l.>,<L.>);/
        (<m>,<M>); (<n>,<N>);/
        (<n'>,<N'>); (<n,>,<N,>);/
        (<n?>,<N?>); (<o>,<O>);/
        (<o!>,<O!>); (<o'>,<O'>);/
        (<o(>,<O(>); (<o->,<O->);/
        (<o1>,<O1>); (<o:>,<O:>);/
        (<o;>,<O;>); (<o?>,<O?>);/
        (<oe>,<OE>); (<p>,<P>);/
        (<q>,<Q>); (<r>,<R>);/
        (<r'>,<R'>); (<r,>,<R,>);/
        (<r<>,<R<>); (<s>,<S>);/
        (<s'>,<S'>); (<s,>,<S,>);/
        (<s<>,<S<>); (<t>,<T>);/
        (<t,>,<T,>); (<t<>,<T<>);/
        (<u>,<U>); (<u!>,<U!>);/
        (<u'>,<U'>); (<u->,<U->);/
        (<u:>,<U:>); (<u;>,<U;>);/
        (<u?>,<U?>); (<v>,<V>);/
        (<w>,<W>); (<x>,<X>);/
        (<y>,<Y>); (<y'>,<Y'>);/
        (<z>,<Z>); (<z'>,<Z'>);/
        (<z<>,<Z<>);

END LC_CTYPE

LC_COLLATE

% Ordering algorithm:
%  1. Spaces and hyphen (but not soft
%     hyphen) before punctuation
%     characters, punctuation characters
%     before numbers,
%     numbers before letters.
%  2. Letters with diacritical marks are
%     members of equivalence classes
%  3. Lower case letters before
%     corresponding upper case letter.
%  4. Specials are ignored when comparing
%     letters, but then they are
%     considered
%  5. The alphabets are sorted in the
%     order of appearance in ISO 10646:
%     Latin, Greek, Cyrillic, etc.

collating-element <s-s> from <s><s>
collating-element <i-j> from <i><j>
collating-element <I-J> from <I><J>
collating-element <o-e> from <o><e>
collating-element <O-E> from <O><E>
collating-element <t-h> from <t><h>
collating-element <T-H> from <T><H>
collating-element <n-g> from <n><g>
collating-element <N-G> from <N><G>

% collating symbols,
% <CAPITAL> or <SMALL> letters first
% <SMALL> letters before <CAPITAL> letters

collating-symbol <SMALL>
collating-symbol <BOTH>
collating-symbol <CAPITAL>

collating-symbol <NONE>
collating-symbol <ACUTE>
collating-symbol <GRAVE>
collating-symbol <CIRCUMFLEX>
collating-symbol <TILDE>
collating-symbol <MACRON>
collating-symbol <BREVE>
collating-symbol <DOT>
collating-symbol <DIAERESIS>
collating-symbol <CEDILLA>
collating-symbol <HORN>
collating-symbol <HOOK>
collating-symbol <UNDERLINE>
collating-symbol <STROKE>
collating-symbol <RING>
collating-symbol <DOUBLE-ACUTE>
collating-symbol <OGONEK>
collating-symbol <CARON>
collating-symbol <GREEK>
collating-symbol <CYRILLIC>
collating-symbol <HIRAGANA>
collating-symbol <KATAKANA>
collating-symbol <ALPHA-1>
collating-symbol <ALPHA-2>
collating-symbol <PRECEDED-BY-APOSTROPHE>
collating-symbol <SPECIAL>
collating-symbol <ACC1>
collating-symbol <ACC3>
collating-symbol <ACC11>
collating-symbol <ACC12>

% letter;accent;case;specials

order_start forward;backward;/
            forward;forward

<CAPITAL>
<BOTH>
<SMALL>

<NONE>
<ACUTE>
<GRAVE>
<CIRCUMFLEX>
<DIAERESIS>
<DOUBLE-ACUTE>
```

```
<CEDILLA>
<OGONEK>
<MACRON>
<STROKE>
<UNDERLINE>
<HOOK>
<TILDE>
<DOT>
<BREVE>
<CARON>
<HORN>
<RING>
<GREEK>
<CYRILLIC>
<ALPHA-1>
<ALPHA-2>
<HIRAGANA>
<KATAKANA>
<PRECEDED-BY-APOSTROPHE>
<SPECIAL>
<ACC1>
<ACC3>
<ACC11>
<ACC12>
<LAST>

<SP> <SP>
<NS> <SP>
<HT> <SP>
<VT> <SP>
<CR> <SP>
<LF> <SP>
<FF> <SP>
<->  <SP>
<//> <SP>
<NU> IGNORE;IGNORE;IGNORE;<NU>
<SH> IGNORE;IGNORE;IGNORE;<SH>
<SX> IGNORE;IGNORE;IGNORE;<SX>
<EX> IGNORE;IGNORE;IGNORE;<EX>
<ET> IGNORE;IGNORE;IGNORE;<ET>
<EQ> IGNORE;IGNORE;IGNORE;<EQ>
<AK> IGNORE;IGNORE;IGNORE;<AK>
<BL> IGNORE;IGNORE;IGNORE;<BL>
<BS> IGNORE;IGNORE;IGNORE;<BS>
<SO> IGNORE;IGNORE;IGNORE;<SO>
<SI> IGNORE;IGNORE;IGNORE;<SI>
<DL> IGNORE;IGNORE;IGNORE;<DL>
<D1> IGNORE;IGNORE;IGNORE;<D1>
<D2> IGNORE;IGNORE;IGNORE;<D2>
<D3> IGNORE;IGNORE;IGNORE;<D3>
<D4> IGNORE;IGNORE;IGNORE;<D4>
<NK> IGNORE;IGNORE;IGNORE;<NK>
<SY> IGNORE;IGNORE;IGNORE;<SY>
<EB> IGNORE;IGNORE;IGNORE;<EB>
<CN> IGNORE;IGNORE;IGNORE;<CN>
<EM> IGNORE;IGNORE;IGNORE;<EM>
<SB> IGNORE;IGNORE;IGNORE;<SB>
<EC> IGNORE;IGNORE;IGNORE;<EC>
<FS> IGNORE;IGNORE;IGNORE;<FS>
<GS> IGNORE;IGNORE;IGNORE;<GS>
<RS> IGNORE;IGNORE;IGNORE;<RS>
<US> IGNORE;IGNORE;IGNORE;<US>
<!>  IGNORE;IGNORE;IGNORE;<!>
<">  IGNORE;IGNORE;IGNORE;<">
<Nb> IGNORE;IGNORE;IGNORE;<Nb>
<DO> IGNORE;IGNORE;IGNORE;<DO>
<%>  IGNORE;IGNORE;IGNORE;<%>
<&>  IGNORE;IGNORE;IGNORE;<&>
<'>  IGNORE;IGNORE;IGNORE;<'>
<(>  IGNORE;IGNORE;IGNORE;<(>
<)>  IGNORE;IGNORE;IGNORE;<)>
<*>  IGNORE;IGNORE;IGNORE;<*>
<+>  IGNORE;IGNORE;IGNORE;<+>
<,>  IGNORE;IGNORE;IGNORE;<,>
<.>  IGNORE;IGNORE;IGNORE;<.>
<:>  IGNORE;IGNORE;IGNORE;<:>
<;>  IGNORE;IGNORE;IGNORE;<;>
<<>  IGNORE;IGNORE;IGNORE;<<>
<=>  IGNORE;IGNORE;IGNORE;<=>
</>> IGNORE;IGNORE;IGNORE;</>>
<?>  IGNORE;IGNORE;IGNORE;<?>
<At> IGNORE;IGNORE;IGNORE;<At>
<<(> IGNORE;IGNORE;IGNORE;<<(>
<_>  IGNORE;IGNORE;IGNORE;<_>
<'!> IGNORE;IGNORE;IGNORE;<'!>
<(!> IGNORE;IGNORE;IGNORE;<(!>
<!!> IGNORE;IGNORE;IGNORE;<!!>
<!)> IGNORE;IGNORE;IGNORE;<!)>
<'?> IGNORE;IGNORE;IGNORE;<'?>
<DT> IGNORE;IGNORE;IGNORE;<DT>
<PA> IGNORE;IGNORE;IGNORE;<PA>
<HO> IGNORE;IGNORE;IGNORE;<HO>
<BH> IGNORE;IGNORE;IGNORE;<BH>
<NH> IGNORE;IGNORE;IGNORE;<NH>
<IN> IGNORE;IGNORE;IGNORE;<IN>
<NL> IGNORE;IGNORE;IGNORE;<NL>
<SA> IGNORE;IGNORE;IGNORE;<SA>
<ES> IGNORE;IGNORE;IGNORE;<ES>
<HS> IGNORE;IGNORE;IGNORE;<HS>
<HJ> IGNORE;IGNORE;IGNORE;<HJ>
<VS> IGNORE;IGNORE;IGNORE;<VS>
<PD> IGNORE;IGNORE;IGNORE;<PD>
<PU> IGNORE;IGNORE;IGNORE;<PU>
<RI> IGNORE;IGNORE;IGNORE;<RI>
<S2> IGNORE;IGNORE;IGNORE;<S2>
<S3> IGNORE;IGNORE;IGNORE;<S3>
<DC> IGNORE;IGNORE;IGNORE;<DC>
<P1> IGNORE;IGNORE;IGNORE;<P1>
<P2> IGNORE;IGNORE;IGNORE;<P2>
<TS> IGNORE;IGNORE;IGNORE;<TS>
<CC> IGNORE;IGNORE;IGNORE;<CC>
<MW> IGNORE;IGNORE;IGNORE;<MW>
<SG> IGNORE;IGNORE;IGNORE;<SG>
<EG> IGNORE;IGNORE;IGNORE;<EG>
```

```
<SS> IGNORE;IGNORE;IGNORE;<SS>
<GC> IGNORE;IGNORE;IGNORE;<GC>
<SC> IGNORE;IGNORE;IGNORE;<SC>
<CI> IGNORE;IGNORE;IGNORE;<CI>
<ST> IGNORE;IGNORE;IGNORE;<ST>
<OC> IGNORE;IGNORE;IGNORE;<OC>
<PM> IGNORE;IGNORE;IGNORE;<PM>
<AC> IGNORE;IGNORE;IGNORE;<AC>
<!I> IGNORE;IGNORE;IGNORE;<!I>
<Ct> IGNORE;IGNORE;IGNORE;<Ct>
<Pd> IGNORE;IGNORE;IGNORE;<Pd>
<Cu> IGNORE;IGNORE;IGNORE;<Cu>
<Ye> IGNORE;IGNORE;IGNORE;<Ye>
<BB> IGNORE;IGNORE;IGNORE;<BB>
<SE> IGNORE;IGNORE;IGNORE;<SE>
<':> IGNORE;IGNORE;IGNORE;<':>
<Co> IGNORE;IGNORE;IGNORE;<Co>
<-a> IGNORE;IGNORE;IGNORE;<-a>
<<<> IGNORE;IGNORE;IGNORE;<<<>
<NO> IGNORE;IGNORE;IGNORE;<NO>
<Rg> IGNORE;IGNORE;IGNORE;<Rg>
<DG> IGNORE;IGNORE;IGNORE;<DG>
<+-> IGNORE;IGNORE;IGNORE;<+->
<''> IGNORE;IGNORE;IGNORE;<''>
<My> IGNORE;IGNORE;IGNORE;<My>
<PI> IGNORE;IGNORE;IGNORE;<PI>
<.M> IGNORE;IGNORE;IGNORE;<.M>
<',> IGNORE;IGNORE;IGNORE;<',>
<-o> IGNORE;IGNORE;IGNORE;<-o>
<14> IGNORE;IGNORE;IGNORE;<14>
<12> IGNORE;IGNORE;IGNORE;<12>
<34> IGNORE;IGNORE;IGNORE;<34>
<?I> IGNORE;IGNORE;IGNORE;<?I>
<*X> IGNORE;IGNORE;IGNORE;<*X>
<-:> IGNORE;IGNORE;IGNORE;<-:>
<'<> IGNORE;IGNORE;IGNORE;<'<>
<'(> IGNORE;IGNORE;IGNORE;<'(>
<'.> IGNORE;IGNORE;IGNORE;<'.>
<'0> IGNORE;IGNORE;IGNORE;<'0>
<';> IGNORE;IGNORE;IGNORE;<';>
<'"> IGNORE;IGNORE;IGNORE;<'">
<-N> IGNORE;IGNORE;IGNORE;<-N>
<-M> IGNORE;IGNORE;IGNORE;<-M>
<'6> IGNORE;IGNORE;IGNORE;<'6>
<'9> IGNORE;IGNORE;IGNORE;<'9>
<.9> IGNORE;IGNORE;IGNORE;<.9>
<"6> IGNORE;IGNORE;IGNORE;<"6>
<"9> IGNORE;IGNORE;IGNORE;<"9>
<:9> IGNORE;IGNORE;IGNORE;<:9>
<%0> IGNORE;IGNORE;IGNORE;<%0>
<1"> IGNORE;IGNORE;IGNORE;<1">
<Ca> IGNORE;IGNORE;IGNORE;<Ca>
<<1> IGNORE;IGNORE;IGNORE;<<1>
<'-> IGNORE;IGNORE;IGNORE;<'->
<Li> IGNORE;IGNORE;IGNORE;<Li>
<Pt> IGNORE;IGNORE;IGNORE;<Pt>
<TM> IGNORE;IGNORE;IGNORE;<TM>
<Om> IGNORE;IGNORE;IGNORE;<Om>
<18> IGNORE;IGNORE;IGNORE;<18>
<38> IGNORE;IGNORE;IGNORE;<38>
<58> IGNORE;IGNORE;IGNORE;<58>
<78> IGNORE;IGNORE;IGNORE;<78>
<-!> IGNORE;IGNORE;IGNORE;<-!>
<-v> IGNORE;IGNORE;IGNORE;<-v>
<==> IGNORE;IGNORE;IGNORE;<==>
<dP> IGNORE;IGNORE;IGNORE;<dP>
<*P> IGNORE;IGNORE;IGNORE;<*P>
<+Z> IGNORE;IGNORE;IGNORE;<+Z>
<Ob> IGNORE;IGNORE;IGNORE;<Ob>
<Sb> IGNORE;IGNORE;IGNORE;<Sb>
<RT> IGNORE;IGNORE;IGNORE;<RT>
<00> IGNORE;IGNORE;IGNORE;<00>
<AN> IGNORE;IGNORE;IGNORE;<AN>
<(U> IGNORE;IGNORE;IGNORE;<(U>
<In> IGNORE;IGNORE;IGNORE;<In>
<?1> IGNORE;IGNORE;IGNORE;<?1>
<?2> IGNORE;IGNORE;IGNORE;<?2>
<!=> IGNORE;IGNORE;IGNORE;<!=>
<=3> IGNORE;IGNORE;IGNORE;<=3>
<=<> IGNORE;IGNORE;IGNORE;<=<>
<.3> IGNORE;IGNORE;IGNORE;<.3>
<NI> IGNORE;IGNORE;IGNORE;<NI>
<Iu> IGNORE;IGNORE;IGNORE;<Iu>
<Il> IGNORE;IGNORE;IGNORE;<Il>
<hh> IGNORE;IGNORE;IGNORE;<hh>
<HH> IGNORE;IGNORE;IGNORE;<HH>
<vv> IGNORE;IGNORE;IGNORE;<vv>
<VV> IGNORE;IGNORE;IGNORE;<VV>
<dr> IGNORE;IGNORE;IGNORE;<dr>
<dR> IGNORE;IGNORE;IGNORE;<dR>
<Dr> IGNORE;IGNORE;IGNORE;<Dr>
<DR> IGNORE;IGNORE;IGNORE;<DR>
<dl> IGNORE;IGNORE;IGNORE;<dl>
<dL> IGNORE;IGNORE;IGNORE;<dL>
<Dl> IGNORE;IGNORE;IGNORE;<Dl>
<LD> IGNORE;IGNORE;IGNORE;<LD>
<ur> IGNORE;IGNORE;IGNORE;<ur>
<uR> IGNORE;IGNORE;IGNORE;<uR>
<Ur> IGNORE;IGNORE;IGNORE;<Ur>
<UR> IGNORE;IGNORE;IGNORE;<UR>
<ul> IGNORE;IGNORE;IGNORE;<ul>
<uL> IGNORE;IGNORE;IGNORE;<uL>
<Ul> IGNORE;IGNORE;IGNORE;<Ul>
<UL> IGNORE;IGNORE;IGNORE;<UL>
<vr> IGNORE;IGNORE;IGNORE;<vr>
<vR> IGNORE;IGNORE;IGNORE;<vR>
<Vr> IGNORE;IGNORE;IGNORE;<Vr>
<VR> IGNORE;IGNORE;IGNORE;<VR>
<vl> IGNORE;IGNORE;IGNORE;<vl>
<vL> IGNORE;IGNORE;IGNORE;<vL>
<Vl> IGNORE;IGNORE;IGNORE;<Vl>
<VL> IGNORE;IGNORE;IGNORE;<VL>
<dh> IGNORE;IGNORE;IGNORE;<dh>
<dH> IGNORE;IGNORE;IGNORE;<dH>
```

```
<Dh> IGNORE;IGNORE;IGNORE;<Dh>
<DH> IGNORE;IGNORE;IGNORE;<DH>
<uh> IGNORE;IGNORE;IGNORE;<uh>
<uH> IGNORE;IGNORE;IGNORE;<uH>
<Uh> IGNORE;IGNORE;IGNORE;<Uh>
<UH> IGNORE;IGNORE;IGNORE;<UH>
<vh> IGNORE;IGNORE;IGNORE;<vh>
<vH> IGNORE;IGNORE;IGNORE;<vH>
<Vh> IGNORE;IGNORE;IGNORE;<Vh>
<VH> IGNORE;IGNORE;IGNORE;<VH>
<TB> IGNORE;IGNORE;IGNORE;<TB>
<LB> IGNORE;IGNORE;IGNORE;<LB>
<FB> IGNORE;IGNORE;IGNORE;<FB>
<lB> IGNORE;IGNORE;IGNORE;<lB>
<RB> IGNORE;IGNORE;IGNORE;<RB>
<.S> IGNORE;IGNORE;IGNORE;<.S>
<:S> IGNORE;IGNORE;IGNORE;<:S>
<?S> IGNORE;IGNORE;IGNORE;<?S>
<fS> IGNORE;IGNORE;IGNORE;<fS>
<Db> IGNORE;IGNORE;IGNORE;<Db>
<Md> IGNORE;IGNORE;IGNORE;<Md>
<"!> IGNORE;IGNORE;IGNORE;<"!>
<"’> IGNORE;IGNORE;IGNORE;<"’>
<"?> IGNORE;IGNORE;IGNORE;<"?>
<"-> IGNORE;IGNORE;IGNORE;<"->
<"(> IGNORE;IGNORE;IGNORE;<"(>
<".> IGNORE;IGNORE;IGNORE;<".>
<":> IGNORE;IGNORE;IGNORE;<":>
<"0> IGNORE;IGNORE;IGNORE;<"0>
<""> IGNORE;IGNORE;IGNORE;<"">
<"<> IGNORE;IGNORE;IGNORE;<"<>
<",> IGNORE;IGNORE;IGNORE;<",>
<";> IGNORE;IGNORE;IGNORE;<";>
<"_> IGNORE;IGNORE;IGNORE;<"_>
<Fl> IGNORE;IGNORE;IGNORE;<Fl>
<UNDEFINED>IGNORE;IGNORE;IGNORE
<0>
<1>  <1>
<1S> <1>
<2>  <2>
<2S> <2>
<3>  <3>
<3S> <3>
<4>
<5>
<6>
<7>
<8>
<9>
<A>  <A>;<NONE>;<CAPITAL>
<a>  <A>;<NONE>;<SMALL>
<A’> <A>;<ACUTE>;<CAPITAL>
<a’> <A>;<ACUTE>;<SMALL>
<A!> <A>;<GRAVE>;<CAPITAL>
<a!> <A>;<GRAVE>;<SMALL>
<A;> <A>;<OGONEK>;<CAPITAL>
<a;> <A>;<OGONEK>;<SMALL>
<A-> <A>;<MACRON>;<CAPITAL>
<a-> <A>;<MACRON>;<SMALL>
<A?> <A>;<TILDE>;<CAPITAL>
<a?> <A>;<TILDE>;<SMALL>
<A(> <A>;<BREVE>;<CAPITAL>
<a(> <A>;<BREVE>;<SMALL>
<A1> <A>;<ACC1>;<CAPITAL>
<a1> <A>;<ACC1>;<SMALL>
<A7> <A>;<ACC7>;<CAPITAL>
<a7> <A>;<ACC7>;<SMALL>
<AA> <A>;<RING>;<CAPITAL>
<AA’>     <A>;<ALPHA-2>;<CAPITAL>
<aa> <A>;<RING>;<SMALL>
<aa’>     <A>;<ALPHA-2>;<SMALL>
<AE> <A><E>;<AE><NONE>;<CAPITAL>
<ae> <A><E>;<AE><NONE>;<SMALL>
<AE’>     <A><E>;<AE><ACUTE>;<CAPITAL>
<ae’>     <A><E>;<AE><ACUTE>;<SMALL>
<A3> <A><E>;<AE><MACRON>;<CAPITAL>
<a3> <A><E>;<AE><MACRON>;<SMALL>
<a8> <A>;<LAST>;<SMALL>
<B>  <B>;<NONE>;<CAPITAL>
<b>  <B>;<NONE>;<SMALL>
<b8> <B>;<LAST>;<SMALL>
<C>  <C>;<NONE>;<CAPITAL>
<c>  <C>;<NONE>;<SMALL>
<C’> <C>;<ACUTE>;<CAPITAL>
<c’> <C>;<ACUTE>;<SMALL>
<C,> <C>;<CEDILLA>;<CAPITAL>
<c,> <C>;<CEDILLA>;<SMALL>
<C<> <C>;<CARON>;<CAPITAL>
<c<> <C>;<CARON>;<SMALL>
<c8> <C>;<LAST>;<SMALL>
<D>  <D>;<NONE>;<CAPITAL>
<d>  <D>;<NONE>;<SMALL>
<d8> <D>;<LAST>;<SMALL>
<E>  <E>;<NONE>;<CAPITAL>
<e>  <E>;<NONE>;<SMALL>
<E’> <E>;<ACUTE>;<CAPITAL>
<e’> <E>;<ACUTE>;<SMALL>
<E!> <E>;<GRAVE>;<CAPITAL>
<e!> <E>;<GRAVE>;<SMALL>
<E:> <E>;<DIAERESIS>;<CAPITAL>
<e:> <E>;<DIAERESIS>;<SMALL>
<E;> <E>;<OGONEK>;<CAPITAL>
<e;> <E>;<OGONEK>;<SMALL>
<E-> <E>;<MACRON>;<CAPITAL>
<e-> <E>;<MACRON>;<SMALL>
<E.> <E>;<DOT>;<CAPITAL>
<e.> <E>;<DOT>;<SMALL>
<E(> <E>;<BREVE>;<CAPITAL>
<e(> <E>;<BREVE>;<SMALL>
<e8> <E>;<LAST>;<SMALL>
<F>  <F>;<NONE>;<CAPITAL>
<f>  <F>;<NONE>;<SMALL>
<fi> <F><I>;<ALPHA-2>;<SMALL>
<fl> <F><L>;<ALPHA-2>;<SMALL>
```

```
<f8> <F>;<LAST>;<SMALL>
<G>  <G>;<NONE>;<CAPITAL>
<g>  <G>;<NONE>;<SMALL>
<G,> <G>;<CEDILLA>;<CAPITAL>
<g,> <G>;<CEDILLA>;<SMALL>
<G<> <G>;<CARON>;<CAPITAL>
<g<> <G>;<CARON>;<SMALL>
<g8> <G>;<LAST>;<SMALL>
<H>  <H>;<NONE>;<CAPITAL>
<h>  <H>;<NONE>;<SMALL>
<h8> <H>;<LAST>;<SMALL>
<I>  <I>;<NONE>;<CAPITAL>
<i>  <I>;<NONE>;<SMALL>
<I’> <I>;<ACUTE>;<CAPITAL>
<i’> <I>;<ACUTE>;<SMALL>
<I!> <I>;<GRAVE>;<CAPITAL>
<i!> <I>;<GRAVE>;<SMALL>
<I:> <I>;<DIAERESIS>;<CAPITAL>
<i:> <I>;<DIAERESIS>;<SMALL>
<I;> <I>;<OGONEK>;<CAPITAL>
<i;> <I>;<OGONEK>;<SMALL>
<I-> <I>;<MACRON>;<CAPITAL>
<i-> <I>;<MACRON>;<SMALL>
<I?> <I>;<TILDE>;<CAPITAL>
<i?> <I>;<TILDE>;<SMALL>
<i.> <I>;<DOT>;<SMALL>
<i8> <I>;<LAST>;<SMALL>
<J>  <J>;<NONE>;<CAPITAL>
<j>  <J>;<NONE>;<SMALL>
<j8> <J>;<LAST>;<SMALL>
<K>  <K>;<NONE>;<CAPITAL>
<k>  <K>;<NONE>;<SMALL>
<K,> <K>;<CEDILLA>;<CAPITAL>
<k,> <K>;<CEDILLA>;<SMALL>
<K<> <K>;<CARON>;<CAPITAL>
<k<> <K>;<CARON>;<SMALL>
<kk> <K>;<ALPHA-2>;<SMALL>
<k8> <K>;<LAST>;<SMALL>
<L>  <L>;<NONE>;<CAPITAL>
<l>  <L>;<NONE>;<SMALL>
<L’> <L>;<ACUTE>;<CAPITAL>
<l’> <L>;<ACUTE>;<SMALL>
<L,> <L>;<CEDILLA>;<CAPITAL>
<l,> <L>;<CEDILLA>;<SMALL>
<L.> <L>;<DOT>;<CAPITAL>
<l.> <L>;<DOT>;<SMALL>
<l8> <L>;<LAST>;<SMALL>
<M>  <M>;<NONE>;<CAPITAL>
<m>  <M>;<NONE>;<SMALL>
<m8> <M>;<LAST>;<SMALL>
<N>  <N>;<NONE>;<CAPITAL>
<n>  <N>;<NONE>;<SMALL>
<N’> <N>;<ACUTE>;<CAPITAL>
<n’> <N>;<ACUTE>;<SMALL>
<N,> <N>;<CEDILLA>;<CAPITAL>
<n,> <N>;<CEDILLA>;<SMALL>
<N?> <N>;<TILDE>;<CAPITAL>
<n?> <N>;<TILDE>;<SMALL>
<’n> <N>;<ALPHA-1>;<SMALL>
<N-G>     <N><G>;<N-G><N-G>;<CAPITAL><CAPITAL>
<n-g>     <N><G>;<N-G><N-G>;<SMALL><SMALL>
<NG> <N><G>;<NG><NG>;<CAPITAL><CAPITAL>
<ng> <N><G>;<NG><NG>;<SMALL><SMALL>
<n8> <N>;<LAST>;<SMALL>
<O>  <O>;<NONE>;<CAPITAL>
<o>  <O>;<NONE>;<SMALL>
<O’> <O>;<ACUTE>;<CAPITAL>
<o’> <O>;<ACUTE>;<SMALL>
<O!> <O>;<GRAVE>;<CAPITAL>
<o!> <O>;<GRAVE>;<SMALL>
<O;> <O>;<OGONEK>;<CAPITAL>
<o;> <O>;<OGONEK>;<SMALL>
<O-> <O>;<MACRON>;<CAPITAL>
<o-> <O>;<MACRON>;<SMALL>
<O?> <O>;<TILDE>;<CAPITAL>
<o?> <O>;<TILDE>;<SMALL>
<O(> <O>;<BREVE>;<CAPITAL>
<o(> <O>;<BREVE>;<SMALL>
<O1> <O>;<ACC1>;<CAPITAL>
<o1> <O>;<ACC1>;<SMALL>
<O-E>     <O><E>;<O-E><O-E>;<CAPITAL><CAPITAL>
<o-e>     <O><E>;<O-E><O-E>;<SMALL><SMALL>
<OE> <O><E>;<OE><OE>;<CAPITAL><CAPITAL>
<oe> <O><E>;<OE><OE>;<SMALL><SMALL>
<o8> <O>;<LAST>;<SMALL>
<P>  <P>;<NONE>;<CAPITAL>
<p>  <P>;<NONE>;<SMALL>
<p8> <P>;<LAST>;<SMALL>
<Q>  <Q>;<NONE>;<CAPITAL>
<q>  <Q>;<NONE>;<SMALL>
<q8> <Q>;<LAST>;<SMALL>
<R>  <R>;<NONE>;<CAPITAL>
<r>  <R>;<NONE>;<SMALL>
<R’> <R>;<ACUTE>;<CAPITAL>
<r’> <R>;<ACUTE>;<SMALL>
<R,> <R>;<CEDILLA>;<CAPITAL>
<r,> <R>;<CEDILLA>;<SMALL>
<R<> <R>;<CARON>;<CAPITAL>
<r<> <R>;<CARON>;<SMALL>
<r8> <R>;<LAST>;<SMALL>
<S>  <S>;<NONE>;<CAPITAL>
<s>  <S>;<NONE>;<SMALL>
<S’> <S>;<ACUTE>;<CAPITAL>
<s’> <S>;<ACUTE>;<SMALL>
<S,> <S>;<CEDILLA>;<CAPITAL>
<s,> <S>;<CEDILLA>;<SMALL>
<S<> <S>;<CARON>;<CAPITAL>
<s<> <S>;<CARON>;<SMALL>
<ss> <S><S>;<ss><ss>;<SMALL><SMALL>
<s-s>     <S><S>;<s-s><s-s>;<SMALL><SMALL>
<s8> <S>;<LAST>;<SMALL>
<T>  <T>;<NONE>;<CAPITAL>
<t>  <T>;<NONE>;<SMALL>
<T,> <T>;<CEDILLA>;<CAPITAL>
```

```
<t,> <T>;<CEDILLA>;<SMALL>
<T<> <T>;<CARON>;<CAPITAL>
<t<> <T>;<CARON>;<SMALL>
<T-H>     <T><H>;<T-H><T-H>;<CAPITAL><CAPITAL>
<t-h>     <T><H>;<T-H><T-H>;<SMALL><SMALL>
<TH> <T><H>;<TH><TH>;<CAPITAL><CAPITAL>
<th> <T><H>;<TH><TH>;<SMALL><SMALL>
<t8> <T>;<LAST>;<SMALL>
<U>  <U>;<NONE>;<CAPITAL>
<u>  <U>;<NONE>;<SMALL>
<U'> <U>;<ACUTE>;<CAPITAL>
<u'> <U>;<ACUTE>;<SMALL>
<U!> <U>;<GRAVE>;<CAPITAL>
<u!> <U>;<GRAVE>;<SMALL>
<U;> <U>;<OGONEK>;<CAPITAL>
<u;> <U>;<OGONEK>;<SMALL>
<U-> <U>;<MACRON>;<CAPITAL>
<u-> <U>;<MACRON>;<SMALL>
<U?> <U>;<TILDE>;<CAPITAL>
<u?> <U>;<TILDE>;<SMALL>
<u8> <U>;<LAST>;<SMALL>
<V>  <V>;<NONE>;<CAPITAL>
<v>  <V>;<NONE>;<SMALL>
<v8> <V>;<LAST>;<SMALL>
<W>  <W>;<NONE>;<CAPITAL>
<w>  <W>;<NONE>;<SMALL>
<w8> <W>;<LAST>;<SMALL>
<X>  <X>;<NONE>;<CAPITAL>
<x>  <X>;<NONE>;<SMALL>
<x8> <X>;<LAST>;<SMALL>
<Y>  <Y>;<NONE>;<CAPITAL>
<y>  <Y>;<NONE>;<SMALL>
<Y'> <Y>;<ACUTE>;<CAPITAL>
<y'> <Y>;<ACUTE>;<SMALL>
<y8> <Y>;<LAST>;<SMALL>
<Z>  <Z>;<NONE>;<CAPITAL>
<z>  <Z>;<NONE>;<SMALL>
<Z'> <Z>;<ACUTE>;<CAPITAL>
<z'> <Z>;<ACUTE>;<SMALL>
<Z<> <Z>;<CARON>;<CAPITAL>
<z<> <Z>;<CARON>;<SMALL>
<z8> <Z>;<LAST>;<SMALL>
<a*> <a*>;<GREEK>;<SMALL>
<b*> <b*>;<GREEK>;<SMALL>
<G*> <G*>;<GREEK>;<CAPITAL>
<D*> <D*>;<GREEK>;<CAPITAL>
<d*> <D*>;<GREEK>;<SMALL>
<e*> <c*>;<GREEK>;<SMALL>
<H*> <H*>;<GREEK>;<CAPITAL>
<m*> <m*>;<GREEK>;<SMALL>
<p*> <p*>;<GREEK>;<SMALL>
<t*> <t*>;<GREEK>;<SMALL>
<S*> <S*>;<GREEK>;<CAPITAL>
<s*> <S*>;<GREEK>;<SMALL>
<*s> <S*>;<SPECIAL>;<SMALL>
<F*> <F*>;<GREEK>;<CAPITAL>
<W*> <W*>;<GREEK>;<CAPITAL>
```

```
order_end

END LC_COLLATE
LC_MONETARY
int_curr_symbol        "SDR "
currency_symbol        "<Cu>"
mon_decimal_point      <,>
mon_thousands_sep      <.>
mon_grouping           3;3
positive_sign          ""
negative_sign          <->
int_frac_digits        2
frac_digits            2
p_cs_precedes          1
p_sep_by_space         0
n_cs_precedes          1
n_sep_by_space         0
p_sign_posn            1
n_sign_posn            1
END LC_MONETARY

LC_NUMERIC
decimal_point          <,>
thousands_sep          <.>
grouping               3;0
END LC_NUMERIC

LC_TIME
abday        "Sun";"Mon";"Tue";"Wed";/
             "Thu";"Fri";"Sat"
day          "Sunday";"Monday";/
             "Tuesday";/
             "Wednesday";/
             "Thursday";/
             "Friday";"Saturday"
abmon        "Jan";"Feb";"Mar";"Apr";/
             "May";"Jun";"Jul";"Aug";/
             "Sep";"Oct";"Nov";"Dec"
mon          "January";"February";/
             "March";"April";"May";/
             "June";"July";"August";/
             "September";"October";/
             "November";"December"
% date formats following to ISO 8601-1988
d_t_fmt      "%Y-%m-%dT%T %Z"
d_fmt        "%Y-%m-%d"
t_fmt        "%T"
am_pm        "";""
t_fmt_ampm   ""
END LC_TIME

LC_MESSAGES
yesexpr      "<<(>YyJjSsOo<)//>.*"
noexpr       "<<(>Nn<)//>.*"
END LC_MESSAGES
```

7. References

REFERENCES

(1) ISO 2375 registration: "International Register of Coded Character Sets to be Used With Escape Sequences", European Computer Manufacturers Association (ECMA), Rue du Rhone 114, CH-1204 Geneve, Switzerland, December 1990.

(2) ISO 2DIS 10646, Information Technology - Universal Multiple-Octet Coded Character Set (UCS), ISO/IEC JTC1/SC2/WG2 N783 (26 December 1991).

(3) ISO/IEC 9945-2.2 CD POSIX Shell and Utilities, informative annex F, ISO/IEC JTC1/SC22 N1063 (October 1991).

(4) IBM National Language Support Reference Manual Volume 2, SE09-8002-01 (March 1990).

(5) IBM 3174 Establishment Controller, Character Set Reference, GA27-3831-02 (March 1990).

(6) IBM 3270 Information Display System Character Set Reference, Chapter 10, GA27-2837-9 (April 1987)

(7) IBM DOS 3.30 Reference (Abridged) 94X9575 (February 1987)

(8) IBM Keyboard layouts and code pages, Part Number 07G4586 (June 1991)

(9) HP LaserJet IIP Printer User's Manual, HP Part No. 33471-90901 (June 1989)

(10) Danish Standard DS 2089, Application of ISO 7-bit coded character set, UDC 681.3:003.62, February 1974.

(11) ISO 2022:1986 Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques.

(12) ISO 6429:1988 Information processing - ISO 7-bit and 8-bit coded character sets - Control functions for 7-bit and 8-bit coded character sets.

(13) Falco 5000 Programmer's Manual, revision A, P/N 210037-000, 1987.

(14) The Unicode Standard Version 1.0 Volume 1, ISBN 0-201-56788-1 (October 1991).

(15) ISO/IEC 9899:1990 Programming Languages - C.

(16) ISO 6937-2:1983 Information Processin - Coded Character sets for text communication - Part 2: Latin alphabethic and non-alphabetic graphic characters.

(17) X/Open Portability Guide, Issue 3, X/Open Company Ltd., Englewood Cliffs, NJ: Prentice-Hall

(18) ISO 4217:1987 Codes for the representation of currencies and funds.