EXTERNAL REFERENCE SPECIFICATION

FOR

INTERIM PASCAL EXTENDED FORMATTER V0.0

Submitted: ____________________
H. A. Wohlwend

Approved: ____________________
P. W. Haynes

____
E. LaRowe

____
R. A. Peterson

REVISION DEFINITION SHEET

REV	DATE	DESCRIPTION
A	04/07/78	Original.
B	05/19/78	Updated to reflect comments received through the DCS review.

1.0 PREFACE

The PASCAL-X source code formatter described in this document is based on past experiences developing and using comparable formatters for the ISWL and SWL languages.

1.1 SCOPE_OF_DOCUMENT

This document is intended to contain information necessary to use the PASCAL Extended formatter in an interactive or batch job.

It is assumed that the reader is familiar with the PASCAL-X language and concepts of the NOS operating system.

1.2 APPLICABLE_DOCUMENTS

PASCAL Extended Language Specification (ARH2298)

2.0 INTRODUCTION

The PASCAL Extended formatter is a utility used to format PASCAL-X source code. It executes as a standalone product.

2.1 PURPOSE

The formatter provides a common tool for formatting PASCAL-X source code prior to compilation. PASCAL-X programs formatted by a common tool will be more consistent, readable and maintainable.

2.2 OVERVIEW

The formatter is executed as an absolute file. Input and output for the formatter are specified with a NOS control card. Formatting may be controlled to a limited extent by the use of PASCAL-X pragmats.

3.0 CONVENTIONS

The formatting conventions are mostly hard coded into the formatter with some conventions being alterable by the use of pragmats.

3.1 INPUT FILE CONVENTIONS

1) Source must be normal PASCAL-X input files (currently the NOS 6/12 character set).

2) Source must be legitimate PASCAL-X source, but is not necessarily a compilation unit. Common decks are formattable.

3) Source file is assumed positioned and local.

4) Only one record on source file is formatted.

5) Source line may be any length as formatter reads to end-of-line.

3.2 FORMATTED OUTPUT FILE CONVENTIONS

1) All lines will be 79 characters or less unless altered by a pragmat.

2) The character set output by the formatter will be identical to the input file character set.

3) The formatted output file is created as a local file and is not rewound before or after.

4) The output file will always start with 1 blank line.

3.3 FORMATTING CONVENTIONS

1) All keywords will be converted to upper case unless altered by pragmat. The predefined data types like integer, char, cell, boolean, string, array, record, and set will always be in lower case.

2) All programmer created identifiers will be converted to lower case unless altered by pragmat.

3) The case of strings and comments will not be altered.

4) Comments starting at the left hand margin will not be formatted.

5) Lines with the asterisk control character in column one are not formatted. These lines are assumed to be maintenance package control statements which will be satisfied before compilation.

6) Blank or empty lines are retained.

7) Blanks are squeezed to 1 except within strings and comments.

8) A ; (semicolon) will cause a new line to be started. If a trailing comment follows on that same line, it will remain there.

9) A space will be added before and/or after certain delimiters. Such as: [and] on attribute lists, the operators, etc.

10) Keywords that start or terminate a structured statement will start a new line.

11) Keywords that form a structured statement will cause following statements to have a margin of 2 greater than the line that contains the KEYWORD. Keywords that terminate a structured statement will decrease this margin by 2. The maximum margin will be 40.

12) If there is no blank line before a VAR, TYPE, CONST, or PROCEDURE statement, or label, one will be put there. A blank line will separate a local variable declaration from the procedure statement list.

3.0 CONVENTIONS
3.3 FORMATTING CONVENTIONS

13) Each parameter declaration will start a new line when they appear within a PROCEDURE declaration. The VAR or value attribute will have a margin two spaces greater than the margin for the PROCEDURE statement. The maximum margin will be 40.

14) Identifiers defined in TYPE, VAR, and CONST statements will each start on a new line.

15) Labels are backed up 2 columns from current margin and are alone on a line.

16) If comments need to be separated across lines, they will be separated at a blank and subsequent lines will be lined up with the original line.

17) If comments starting at the left hand margin are too long, they are continued at the left hand margin of succeeding lines until the comment is completed.

18) The left hand column defaults to 1 unless altered by a pragmat. The starting margin is equal to the left hand column.

    NOTE: The first input line is given a margin of 1.

19) Pragmats that are processed by the formatter take effect after their appearance in the source.

20) The formatter checks column one of the input stream and if it is non-numeric or the asterisk control character, the information is assumed to be text.

21) The formatter stops execution and outputs an appropriate message upon detecting the first syntax error. The status of the output file is undefined.

22) The THEN clause of the IF statement will normally be on the same line as the IF, space permitting.

23) The ELSE and IFEND clauses are aligned with the IF statement.

24) CASE selection specs will have the same margin as their associated CASE statement.

25) Statements that exceed the length of a line will be continued on the following line with a 2 character

indentation.

26) Long strings that do not fit on a single line will be split and the CAT operator generated.

3.4 SUGGESTIONS

The following list of formatting features will not be done by the interim formatter, but should be strongly considered for the ultimate PASCAL-X formatter.

o Formatting of PLs.

o Filling in the labels on ending delimiters of labeled structured statements, procends and modends.

o Verifying that if the 4th character of a name is a $ sign, then the 3rd character must conform to the C180 System Interface Standard.

o Enforcing C180 coding conventions as they are defined. This includes capitalization of keywords and user defined identifiers.

o Recognition and justification of comments within a comment block.

o Develop a method of highlighting changes in flow of execution caused by the EXIT, CYCLE and RETURN statements.

o Investigate readability of tabbing the ':' in a type, variable, field or parameter specification to a fixed position.

4.0 PRAGMATS

Pragmats appearing in the source are used to control the PASCAL-X formatter and will be copied to output to provide a graceful means of reformatting.

4.1 PASCAL-X PRAGMATS

These PASCAL-X pragmats control the PASCAL-X compiler as well as the formatter.

4.1.1 ??RIGHT := N??

For the PASCAL-X formatter, this pragmat is used to specify maximum output line length with n >= 72 and <= 110. The default value is 79. This pragmat indicates to the PASCAL-X compiler the maximum length of a source line.

4.1.2 ??LEFT := N??

Used to make the PASCAL-X compiler and formatter ignore any foreign data outside the PASCAL-X syntax. For the formatter, this pragmat indicates left hand output column and starting margin. The default value is 1. The foreign data which exists in the ignored column positions will be copied to the output file.

4.2 FORMATTER PRAGMAT

This pragmat controls only the PASCAL-X formatter and is not processed by the PASCAL-X compiler.

4.2.1 SYNTAX

```
<formatter pragmat statement> ::= ??<formatter pragmat>??

<formatter pragmat> ::= FMT (<toggle setting list>)

<toggle setting list> ::= <toggle setting> [,<toggle setting>]

<toggle setting> ::= <toggle identifier> := <condition>

<toggle identifier> ::= FORMAT ! KEYW ! IDENT

<condition> ::= ON ! OFF ! UPPER ! LOWER
```

4.2.2 TOGGLE IDENTIFIERS

o FORMAT

Used to control formatting of all lines. Default is FORMAT := ON.

FORMAT := ON - All source lines are formatted.

FORMAT := OFF - No source lines are formatted.

o) KEYW

Used to control capitalization of keywords. Default KEYW := UPPER.

KEYW := UPPER - All keywords will be upper case.

KEYW := LOWER - All keywords will be lower case.

o IDENT

Used to control capitalization of identifiers. Default IDENT := LOWER.

IDENT := UPPER - All identifiers will be upper case.

IDENT := LOWER - All identifiers will be lower case.

5.0 USE_OF_FORMATTER

The PASCAL-X formatter is executed as an absolute file.

```
ATTACH,PASFORM/UN=LP3
PASFORM[(<I=filename>][<,O=filename>)]
```

I:	Specifies the name of the file to be formatted. If I is not specified, the name of the file is I.
O:	Specifies the name for the formatted file. If O is not specified, the name of the file is O.

NOTE: I and O may not use the same file name.

6.0 EXAMPLES

6.1 SOURCE_PROGRAM

1)
```
          ??RIGHT := 110??
    MODULE M1;
    PROCEDURE A1;
    VAR I: INTEGER;
    PROCEND A1;
    MODEND M1;
```

Formatted is:

```
    ??RIGHT := 110??
    MODULE M1;

      PROCEDURE A1;

        VAR
          I: INTEGER;
      PROCEND A1;
    MODEND M1;
```

2)
```
          ??RIGHT := 72??
      MODULE M1;
         PROCEDURE A1;
           VAR
             I: INTEGER;
         CONST J=5;
         PROCEND A1;
      MODEND M1;
```

Formatted is:

```
    ??RIGHT := 72??
    MODULE M1;

      PROCEDURE A1;

        VAR
          I: INTEGER;
```

6.0 EXAMPLES
6.1 SOURCE PROGRAM

```
        CONST
          J=5;
      PROCEND A1;
    MODEND M1;
```

6.2 CONTROL CARDS

```
GET,I=SOURCE.
ATTACH,PASFORM/UN=LP3.
PASFORM.
REPLACE,O=COMPILE.
```

The file COMPILE will contain the formatted program found on file SOURCE.

6.3 MESSAGES

Messages are written to the OUTPUT file and the dayfile. The source line that caused the error message will normally be the last one on the output file.

1) ***UNBALANCED BLOCK STRUCTURE***

   Begin and end statements for structured statements do not match.

2) ***IMPROPER PARAMETER LIST***

   Something is wrong with a PROCEDURE definition statement.

3) ***EXTRANEOUS RIGHT PARENTHESIS***

   Right and left parens do not match.

4) ***MISSING RIGHT PARENTHESIS***

   Right and left parens do not match.

5) ***IMPROPER CASE LABEL***

   Improper case statement.

6) ***IMPROPER PRAGMAT***

Something is wrong with a pragmat that the PASCAL-X formatter processes.

7) ***IMPROPER HEX CONSTANT***

Something is wrong with a hex constant.

8) ***IDENTIFIER TOO LONG***

Have an identifier greater than 31 characters.

9) ***IMPROPER STRING CONSTANT***

A quote is missing at EOL causing a syntactic error.

Table of Contents