3 PHASES of implementation

1) = Wirth Compiler
2) = Wirth Compiler + OS/COMMS extensions
3) = Full compiler (as defined in PE-TT-1145 rev 2)
PHASE 1): Use Wirth book as a reference guide

Extensions to Wirth included in PHASE 1:

DEFINITION MODULE EXPORT statement:

Use EXPORT QUALIFIED to ensure Modula-Modula name resolution

Don't use EXPORT QUALIFIED to ensure Modula-other or other-Modula name resolution

ENUMERATIONS:

27 enumeration = "(" enumeration_element_list ")".
27a enumeration_element_list = enumeration_element {"," enumeration_element}.
27b enumeration_element = ident [initial_value].
27c initial_value = "=" ConstExpression.

Where ConstExpression should evaluate to an INTEGER or CARDINAL value.

SUBRANGES:

29 SubrangeType = [qualident] "[" ConstExpression "] .." ConstExpression "]".

Where qualident has the same semantic meaning as in set constants.

LETTERS:

1 ident = letter {letter | digit | "." | ";"}. 


Compiler defaults for Alpha-Test

-Binary -Optimize 0 -Upcase -Errtty -Map

use -HELP to get help about command line options

UPCASE:

will upcase RESERVED WORDS and IDENTs (including QUALIDENTs)

Always upcased:

 Exponents: E, D, X, Q (as implemented)

 Constant tags: B, C, H

 Hex digits A, B, C, D, E, F
This Installation: 

ALPHA-TEST compiler!

WITH statements: Not implemented yet...

[ low .. high ]: Don't work (Type them for now...)

enumerations: Aren't checked for proper ordering (yet...)

RECORDS: Not tested, probably don't work yet...

ARRAYs: Not tested, probably don't work yet...

Procedure Types: Not implemented, DON'T work yet...

Variant RECORDS: Not implemented, DON'T work yet...

IMPORT/EXPORT: Not implemented, DON'T work yet...

DEFINITION MODULES: Not implemented, DON'T work yet...

Predeclared names: All Types implemented except BITSET.

No "importable" names (ADR, NEW, DISPOSE, etc.)

QUALIFIED idents: Don't until RECORD and/or IMPORT/EXPORT work.
Extension support by Phase

<table>
<thead>
<tr>
<th>Section</th>
<th>Description of extension</th>
<th>Phase implemented in</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>&quot;Prime extensions&quot; compiler flag</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>-UCASE/LCASE</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>-CASE/NO_CASE</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>256 character identifiers</td>
<td>1</td>
</tr>
<tr>
<td>3.2</td>
<td>Unrepresentable Numbers</td>
<td>as implemented</td>
</tr>
<tr>
<td>3.3</td>
<td>String support</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>escape characters</td>
<td>2</td>
</tr>
<tr>
<td>3.4</td>
<td>Operator equivalences</td>
<td>1</td>
</tr>
<tr>
<td>3.6</td>
<td>Compiler directives</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Array and Record constants</td>
<td>2</td>
</tr>
<tr>
<td>6.1.1</td>
<td>(1, 2) INTEGER, CARDINAL, SHORT_INTEGER, SHORT_CARDINAL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>INTEGER_x, CARDINAL_x</td>
<td>2</td>
</tr>
<tr>
<td>6.1.4</td>
<td>CHAR types ASCII_x, EBCDIC, etc.</td>
<td>3</td>
</tr>
<tr>
<td>6.1.5</td>
<td>REAL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>DOUBLE, QUAD</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>EXTENDED, IEEE</td>
<td>3</td>
</tr>
<tr>
<td>6.1.6</td>
<td>DECIMAL</td>
<td>3</td>
</tr>
<tr>
<td>6.2</td>
<td>Enumerations with holes</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Non-unique Enumerated constants</td>
<td>2</td>
</tr>
<tr>
<td>6.3</td>
<td>Typed Subranges</td>
<td>1</td>
</tr>
<tr>
<td>6.6</td>
<td>Set size</td>
<td>1</td>
</tr>
</tbody>
</table>
6.7 LONG POINTER
   Pointer visibility (type overlays)  2
6.10 Strings  2
8.2.4 IEEE comparisons  3
8.2.5 String concat: +  2
9.2 Pointer valued designators  2
9.5 CASE ELSE missing & range support  2
9.8 IN iterator for FOR statement  2
9.11 required RETURNS in functions  2
10 INLINE procedures  3
   SHORTCALL procedures  2
   GATE procedures  2
10.1 CONST Parameters  3
   Parameterized Arrays  2
   OPTIONAL parameters  3
10.2 EVEN, UNCAP  2
   OS extended BIFs/BICs ala SH/PLP  2
12 System dependant facilities  as implemented
13 Wirth Processes  3
14 External Names  1
15 Exception Handling  2
16 -RANGE support  2
17 Annotations (level 0 errors)  as implemented
18 Conditional Compilation  3