Block = *LabelDeclarationPart ConstantDefinitionPart TypeDefinitionPart*
VariableDeclarationPart ProcedureAndFunctionDeclarationPart
StatementPart .

The label declaration part introduces zero or more labels, each of which must prefix one statement in the statement part.

LabelDeclarationPart = ["`label`" *DigitSequence* ["," *DigitSequence*] ";"] .
Label = *DigitSequence* .

The *spelling* of a label is the apparent integral value that its digit sequence describes in the usual decimal notation; the value must not exceed 9999.

10.2. Scope

A definition or declaration introduces a spelling of an identifier or a label and associates the spelling with a specific meaning (e.g., a variable identifier). The parts of a program in which every occurrence of that spelling must take on that meaning are collectively called the *scope* of the introduction (definition or declaration). The occurrence of a spelling in its introduction must precede every other occurrence of that spelling within the scope of the introduction, with one exception. The exception is that a type–identifier spelling may occur as the domain type of a pointer type (see Section 6.3) anywhere in the type definition part that contains the spelling's introduction.

Each introduction is effective for some region of the program, as described below. The scope of the introduction is that region less any enclosed region for which another introduction of the same spelling is effective.

The following introductions are effective for the block in which the introduction occurs: a label in a label declaration part; a constant identifier in a constant definition part or in an enumerated type; a type identifier in a type definition part; a variable identifier in a variable declaration part; a procedure identifier in a procedure declaration (see Section 11.1); and a function identifier in a function declaration (see Section 11.2). These labels and identifiers are said to be *local* to the block.

The implicit introduction of standard predefined and predeclared