**ALGEBRAIC OPERATORS**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Operand Type</th>
<th>Result Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
<td>I or R</td>
<td>I or R</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
<td>I or R</td>
<td>I or R</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
<td>I or R</td>
<td>I or R</td>
</tr>
<tr>
<td>/</td>
<td>REAL division</td>
<td>I or R</td>
<td>R</td>
</tr>
<tr>
<td>DIV</td>
<td>INTEGER division</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>MOD</td>
<td>Modulus (A MOD B)</td>
<td>I</td>
<td>I</td>
</tr>
</tbody>
</table>

* Assigns value to

* I = INTEGER, R = REAL

**RELATIONAL OPERATORS**

- Equal
- < Not equal
- < Less than
- < Greater than
- < Less than or equal
- > Greater than or equal
- NOT Logical "Not"
- AND Logical "And"
- OR Logical "Or"
- IN SET membership

**PROGRAM STRUCTURE**

```plaintext
PROGRAM ProgramName;
   Declarations
```

Declares name of program

```plaintext
PROCEDURE Proc1Name;
   Declarations
```

Declares name of a procedure

```plaintext
FUNCTION Func1Name;
   Declarations
```

Declare name of a function

**STANDARD (BUILT-IN) IDENTIFIERS**

**Constants**

- FALSE and TRUE Boolean values
- MAXINT Maximum integer value

**Types**

The types with an asterisk (*) are available in UCSD Pascal:

- BOOLEAN
- CHAR
- INTEGER
- REAL
- STRING

**FUNCTIONS**

**Numeric Functions**

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter Type</th>
<th>Result Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS(x)</td>
<td>I or R</td>
<td>Same as operand</td>
<td>Returns absolute value of x</td>
</tr>
</tbody>
</table>
ATAN(x) or ARCTAN(x) I or R R Returns the inverse tangent of x in radians
COS(Angle) I or R R Returns the cosine of Angle
EXP(x) I or R R Returns e to the xth power (e^x)
LN(x) I or R R Returns the natural logarithm of x (x must be greater than 0)
LOG(x) I or R R Returns the Logarithm to the base 10 of x
ROUND(x) R I Round off x to the nearest integer
SIN(Angle) I or R R Returns the sine of Angle
SQR(x) I or R R Returns x squared (x^2)
SQRT(x) I or R R Returns the square root of x (x must be positive)
TRUNC(x) R or L I Converts x to integer without rounding

* I = INTEGER, R = REAL, L = LONG INTEGER

Ordinal Functions

Name | Parameter Type* | Result Type* | Description
--- | --- | --- | ---
ORD(x) | O | I | Returns the position which x holds in its data type
PRED(x) | O | Same as param | Returns the predecessor of x
SUCC(x) | O | Same as param | Returns the successor of x

* I = INTEGER, O = Ordinal
\* if none exists, there will be an error

Other Functions

Name | Parameter Type* | Result Type* | Description
--- | --- | --- | ---
CHR(x) | I | CHR | Returns a character which has the ASCII value x
ODD(x) | I | BOOLEAN | Returns TRUE if x is odd, otherwise returns FALSE

* I = INTEGER

String Functions and Procedures

In the following String intrinsics, the parameters StartPos, Pos and Size are INTEGERS. All other parameters are STRINGS.

Name | Result Type* | Description
--- | --- | ---
CONCAT(Startstr, Str2, ..., StrN) | S,F | Returns a new string which is the concatenation of Str1 through StrN

COPY(SourceStr, StartPos, Size) | S,F | Copies from SourceStr beginning at StartPos taking Size characters
DELETE(SourceStr, StartPos, Size) | P | Removes Size characters from SourceStr beginning at StartPos
INSERT(Source, Dest, Pos) | P | Inserts Source into Dest at Pos
LENGTH(STR) | 1,F | Returns the length of Str
POS(Pattern, SourceStr) | 1,F | Returns the position of the first occurrence of Pattern in SourceStr
CONVERTS(x, either an I or a LONG INTEGER) to a STRING | P | Result is assigned to DestStr

* I = INTEGER, S = STRING, F = Function, P = Procedure

COMMANDS

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE</td>
<td>Use when you want to select one of many statements to execute. The statement following the constant which matches the value of the case-index is executed. Constant list is a list of constants separated by commas.</td>
</tr>
<tr>
<td>EXIT</td>
<td>Use to prematurely leave a procedure or function.</td>
</tr>
<tr>
<td>FOR</td>
<td>Use when you want to repeat a statement(s) a specific number of times.</td>
</tr>
<tr>
<td>IF-THEN</td>
<td>Use when you want to execute a statement(s) only if a specific condition is true.</td>
</tr>
<tr>
<td>IF-THEN-ELSE</td>
<td>Use when you want to execute one of two statements.</td>
</tr>
<tr>
<td>REPEAT-UNTIL</td>
<td>Use when you want to repeat a statement(s) until a specific condition is true. Statement will execute at least once.</td>
</tr>
<tr>
<td>WHILE</td>
<td>Use when you want to repeat a statement(s) only while a specific condition is true. Statement(s) may not execute at all if condition starts out false.</td>
</tr>
</tbody>
</table>

RESERVED WORDS

The words with an asterisk (*) following them are not covered in this book:

AND | ELSE | MOD | RECORD | VAR
--- | --- | --- | --- | ---
ARRAY | END | NIL* | REPEAT | WHILE
BEGIN | FILE* | NOT | SET | WITH*
CASE | FOR | OF | THEN |
CONST | FUNCTION | OR | TO |
DIV | GOTO* | PACKED* | TYPE |
DO | IF | PROCEDURE | UNTIL |
DOWNTO | LABEL* | PROGRAM | USES |

FLOW OF CONTROL COMMANDS

In the following examples, any statement may be substituted by a Compound Statement.