# Rietveld Basic

## 1. *Data types:*

| | |
|---|---|
| BOOLEAN | true/false |
| INTEGER | -2147483648 .. 2147483647 |
| SINGLE | $1.5 * 10^{-45}$ .. $3.4 * 10^{38}$ (7 significant figures) |
| DOUBLE | $5.0 * 10^{-324}$ .. $1.7 * 10^{308}$ (15 significant figures) |
| STRING | up to 255 characters |
| DATE | date |

One dimension arrays of the above
(INTEGER and BOOLEAN types are equivalent. Boolean has the following means: 1 = true, 0 =false)

*Examples:*

```
DIM    A[10]                  // single array A
DOUBLE F, W
STRING s1, s2, s3, s4
STRING s5[10]                 // string array s5
```

## 2. *Operations:*

| | |
|---|---|
| + | addition |
| - | subtraction |
| * | multiplication |
| / | division |
| < | less than |
| <= | less than or equal |
| >= | greater than or equal |
| > | greater than |
| == | equivalence |
| = | assign |
| <> | not equal |
| OR | OR operator |
| AND | AND operator |
| () | function |
| [] | array |

*Examples:*

```
Boolean b1,b2,test
B1 = true
B2 = false
Test = B1 or B2
If test == true then              /can also use: 'If test then'
  Write('this will print')
End if
```

## 3. *Functions:*

### *Mathematical functions*

ROUND(V as single)  as single
FLOOR(V as single)  as single
TRUNC(V as single)  as single

SQR(V as single)  as single
SQRT(V as single)  as single
SIN(V as single)  as single
COS(V as single)  as single
TAN(V as single)  as single
ARCSIN(V as single)  as single
ARCCOS(V as single)  as single
ARCTAN(V as single)  as single
LN(V as single)  as single
LOG2(V as single)  as single
LOG10(V as single)  as single
LOGN(BASE as single, V as single) as single
POWER(VAR as single, P as single) as single
EXP(V as single)  as single
ABS(V as single)  as single

### *String functions*

GetLength(S as string) as integer
SetLength(ByRef S as string, N as integer) as string
StrCopy(S as string, Index as integer, Count as integer) as string   // Return Count bytes from string S from the position Index
StrSet(ByRef S1 as string, Index as integer, S2 as string) as string   // copying string S2 to the string S1 from the position Index
StrNSet(ByRef S1 as string, Index as integer, Count as integer,  S2 as string) as string  // copying Count characters of the string S2 to the string S1 from the position Index
StrUpper(S as string) as string
StrLower(S as string) as string

### *Date functions*

EncodeDate(Y as Integer, M as Integer, D as Integer ) as Date
DecodeDate(D as Date, ByRef Year as integer, ByRef Month as integer, ByRef Day as Integer ) as Integer

### *Data converting function*

FormatVal(FormatStr as string, Val as single ) as string
Val(S as string) as single
Str(Val as single) as string
DateToStr(D as date) as string
StrToDate(S as string) as date

*Examples:*

```
STRING s1, s2, s3, s4

s1 = 'Test string'
s2 = StrCopy( s1, 6, 6 )   // s2 = “string”
StrSet(s3, 3, "basic")     // s3 = “  basic”
StrNSet(s4, 3, 3, "basic") // s4 = “  bas”
```

### *Input/Output functions*

Beep()
Write(any 1 variable)
ClearScr()

Inputbox(s as string) as string
MessageBox(s as string, s2 as string)
Opendialog(S1 as string, S2 as string) as string
SaveDialog(S1 as string, S2 as string) as string

## *File functions*

GetCurrentDir() as string
SetCurrentDir(s1 as string)
CopyFile(s1 as string, s2 as string)
RenameFile(s1 as string, s2 as string)
MoveFile(s1 as string, s2 as string)
DeleteFile(s1 as string)

## *Rietveld functions*

Refinefile(filename as string) as integer
OpenRietveld(filename as string) as integer
StartRietveld() as integer
StepRietveld(#ofsteps as integer) as integer
EndRietveld() as integer
GetParameter(histno, phaseno,atomno,varno as integer) as single
SetParameter(histno, phaseno,atomno,varno as integer, value as single) as integer
GetError(histno as integer) as string
GetFit(histno as integer) as single
PlotRefine(histno as integer)

Where histno is the histogram number, phaseno is the phase number, atomno is the atom number in a particular phase, varno is the variable number specified:

If histno = histogram number and phaseno = 0, atomno=0 then varno defines the following
1 = zero
2..13 = Background values
14 = Histogram Scale
15 = Wavelength 1 or DifC
16 = DifA

if phaseno = phase number and histno = 0,atomno=0 then varno takes the following
1 = Phase Scale
2 = Isotropic Thermal
6 = a
7 = b
8 = c
9 = Alpha
10 = Beta
11 = Gamma

if phaseno = phase number and histno= the histogram number and atomno=0 then varno takes the following
3=U
4=V
5=W
12=Preffered Orientation
13=R value/Flat Plate P0
14=Asymmetery
15=Gam-0
16=Gam-1

17=Gam-2
18=Extinction
19=Uaniso
21=TOF Alpha-1
22=TOF Alpha-2
23=TOF Beta-1
24=TOF Beta-2
25=Flat Plate Pore
26=Flat Plate Rough

if phaseno = phase number atomno=the atom number and histno =0 then varno takes the following

```
1:   hns := 'x';
2:   hns := 'y';
3:   hns := 'z';
4:   hns := 'B';
5:   hns := 'n';
6:   hns := 'B11';
7:   hns := 'B22';
8:   hns := 'B33';
9:   hns := 'B12';
10:   hns := 'B13';
11:   hns := 'B23';
```

if histno=the histogram number,phaseno = 99 then
atomno = 1 for f'
atomno = 2 for f''
and varno is the scattering set number.

## ***4. Constructions:***

### *For-Next Loop*

```
FOR  variable = expression1 TO expression2 [STEP expression3]
…..body…..
NEXT [variable]
```

### *Do-Until Loop*

```
DO
…..body…..
LOOP UNTIL expression1
```

### *Do While Loop*

```
DO WHILE expression1
…..body…..
LOOP
```

### *If-Then*

```
IF expression1 THEN
…..body…..
[ELSEIF expression2]
…..body…..
[ELSEIF expression3]
…..body…..
```

```
 [ELSE]
…..body…..
END IF
```

### *Select Case*

```
SELECT CASE variable
CASE expression1 [ , expression2 [ ,…] ]
…..body…..
CASE expression3 [ , expression4 [ ,…] ]
…..body…..
CASE ELSE
…..body…..
END SELECT
```

### *Functions*

```
FUNCTION FuncName [ ( ParamList ) ] as [ TypeName ]
  FuncName = expression1
…..body…..
  EXIT FUNCTION
  FuncName = expression2
…..body…..
END FUNCTION

// ParamList:
//      [ByRef|ByVal] ParamName1 [as TypeName ], …
```

*Example:*

```
function F1(ByVal N as integer) as integer
  if N > 0 then
    F1 = N * F1( N-1 )
  else
    F1 = 1
  end if
end function
```

*TypeName - can be one of the following:*

```
INTEGER
SINGLE
DOUBLE
STRING
DATE
BOOLEAN
// A function can be called recursive.
```

### *Goto*

```
GOTO LabelName
…
LabelName:
…
```

## *Class*

```
CLASS ClassName [ ( ParentClass ) ]
Type1 variable1
Type2 variable2
…
FUNCTION F1[ ( ParamList ) ] as Type3
…
END FUNCTION
FUNCTION FN[ ( ParamList ) ] as TypeN
…
END FUNCTION
END CLASS

// Type1 .. TypeN - any valid type
```

## *Comments*

// - 'C++'- style comments