Chapter 5 – Decisions

- 5.1 Relational and Logical Operators
- 5.2 If Block

5.1 Relational and Logical Operators

- ANSI Values
- Relational Operators
- Logical Operators
- Boolean Data Type

ANSI Character Set

- A numeric representation for every key on the keyboard and for other assorted characters.

32 (space)	48 0	66 B	122 z
33 !	49 1	90 Z	123 {
34 “	57 9	97 a	125 }
35 #	65 A	98 b	126 ~

ANSI Character Set: continued

- A numeric representation for every key on the keyboard and for other assorted characters.

162 ¢	177 ±	181 µ	190 ¼
169 ©	178 ²	188 ¼	247 ÷
176 °	179 ³	189 ½	248 ø

Chr Function

For n between 0 and 255,

`Chr(n)`

is the string consisting of the character with ANSI value *n*.

EXAMPLES:

```
Chr(65) is "A"
Chr(162) is "¢"
```

Asc Function

For a string *str*,

`Asc(str)`

is ANSI value of the first character of *str*.

EXAMPLES:

```
Asc("A") is 65
Asc("¢25") is 162
```

Relational Operators

<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
=	equal to
<>	not equal to

ANSI values are used to decide order for strings.

Boolean Data Type

- An expression or variable that evaluates to either True or False is said to have Boolean data type.

- *Example:*

  The statement

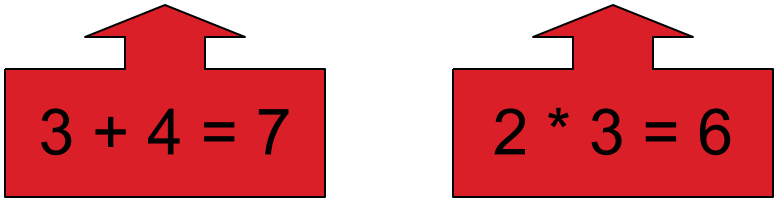
  ```
  txtBox.Text = (2+3)<6
  ```

  displays **`True`** in the text box.

Example

When a = 3, b = 4

(a + b) < **2 * a**



7 is NOT less than 6 and the value of the expression is False

Another Example

a = 4 b = 3 c = "hello" d = "bye"

(c.Length – b) = (a / 2)

True because 2 equals 2

Relational Operator Notes

- Relational operators are binary – they require an operand on both sides of the operator
- Value of a relational expression will always be True or False
- Expressions are evaluated from left to right with no order of operations

Logical Operators

- Used with Boolean expressions
- *Not* – makes a False expression True and vice versa
- *And* – will yield a True if and only if both expressions are True
- *Or* – will yield a True if at least one of both expressions are True

Example

To test if *n* falls between 2 and 5:

(2 < n) And (n < 5)

A complete relational expression must be on either side of the logical operators And and Or.

Syntax error

The following is NOT a valid way to test if *n* falls between 2 and 5:

(2 < n < 5)

Example 5.3

n = 4, *answ* = “Y”

Are the following expressions true or false?

Not (n < 6)

(answ = "Y") Or (answ = "y")

(answ = "Y") And (answ = "y")

Not(answ = "y")

Order of Operations

The order of operations for evaluating Boolean expressions is:

1. Arithmetic operators
2. Relational operators
3. Logical operators

Arithmetic Order of Operations

1. Parenthesis
2. Exponentiation
3. Division and multiplication
4. Addition and subtraction

Relational Order of Operations

They all have the same precedence

Logical Order of Operations

1. Not
2. And
3. Or

Condition

- A *condition* is an expression involving relational and/or logical operators
- Result of the condition is Boolean – that is, True or False

Common Error in Boolean Expressions

- A common error is to replace the condition *Not (2 < 3)* by the condition (2 > 3)

- The correct replacement is *(2 >= 3)*
  because >= is the opposite of <,
  ` just as <= is the opposite of >

Boolean Variable

A variable declared with a statement of the form

```
Dim var As Boolean
```

is said to have Boolean data type. It can assume just the two values True and False.

Example:

```
Dim boolVar As Boolean
boolVar = 2 < 6
txtBox.Text = boolVar
```

displays **`True`** in the text box.