



Delphi

Contents

IF Statements:	2
Compound Statements:	2
Nested Ifs:	3
Logical Ifs:	4
Case Statement:	5
LOOPS:	6

Delphi Chapter 4:

IF Statements:

The IF statement is an essential part to any code language, without it comparisons or conditions can never be used to execute a desired outcome dependent on the conditions. Within Delphi there are 3 types of IF statements that we will cover.

Compound Statements:

This is the most basic of all IF statements including an IF, Then and Else. Also note that the Else statement is only required if there is an alternate outcome to the IF Condition if that condition is not met.

Example:

```

• procedure TFrmIF.BtnInputClick(Sender: TObject);
• begin
28 Name:= Uppercase(Inputbox('Enter', '"Yes"', 'Here'));
•   If Name = 'YES'
30     Then FrmIF.Caption:= 'True'
•     Else FrmIF.Caption:= 'False'
• end;
•
• end.

```

//This is not the greatest example of a Nested IF but it does demonstrate the possibly power of one.

As an algorithm this can be written as:

```

IF <Condition>
  Then <True choice executed>
  Else <False choice executed>

```

Or

```

IF <Condition>
  Then begin
    <True choice executed>
    <Additional Statements >
  End
  Else begin
    <False choice executed>
    <Additional Statements >
  End

```

Nested Ifs:

The Nested IF statement is used if there is specific alternative to the true statement, therefore there is the use of an Else IF. There is no specific limit to the length of nested if, but is usually determined by the amount of alternate but specific conditions. However Logical Ifs and Case Statements are preferred to rather than very long Nested IF.

Example:

```
procedure TFrmIfs.BtnGenderClick(Sender: TObject);
var
  Gender: String;
begin
  30  Gender:= inputbox('Please Enter Gender:', 'Male or Female', 'Male');
  If Lowercase(Gender)= 'male'
  Then Showmessage('Um.. ya ur a : MAN')
  Else If Lowercase(Gender)= 'female'
  Then Showmessage('Um.. ya ur a : FEMALE')
  Else begin
    Showmessage('Um.. Please re-enter gender');
    BtnGenderClick(Sender)
  end;
end;
40  end.
end.
```

//This is not the greatest example of a Nested IF but it does demonstrate the possibly power of one.

As an algorithm this can be written as:

```
IF <Condition1>
  Then <Statement1>
  Else IF <Condition2>
    Then <statement2>
    Else <statement3>
```

Logical Ifs:

The Logical IF statement is used when there is more than one possible condition that will execute the same statement if both or one of the conditions is met. This is the reason for an or/and, dependant on whether both conditions must be met or only one.

Example:

```
procedure TFrmIfs.BtnGenderClick(Sender: TObject);
var
  Name: String;
  Gender: String;
30 begin
  name:= inputbox('Please Enter:', 'Your Name', 'Jeremy');
  Gender:= inputbox('Please Enter Gender:', 'Male or Female', 'Male');
  If (Lowercase(Gender)= 'male') or (Lowercase(Gender)= 'Female')
  Then showmessage('Welcom '+Name+' You are a '+Gender)
  Else BtnGenderClick(Sender);
end;
end.
```

//This is not the greatest example of a Nested IF but it does demonstrate the possibly power of one.

As an algorithm this can be written as:

```
IF <Condition1> AND/OR <Condition 2>
  Then <True choice executed>
  Else <False choice executed>
```

NB: In Delphi there is no semicolon (;) before the else.

Case Statement:

The case Statement is used as a comparison between certain values of a variable to determine another variables value. This can be used to assign a symbol determined by a mark as a percentage. E.G: 70-79 = B. The Case Statement is ideal to avoid long redundant If statements.

Example: *This is a perfect example to show how a case statement can be used to assign a Level determined by how much Experience a player has in a simple RPG.*

```

30  * //Calculates character's or selection's Level using their EXP.
  * //May be moved later to "PROCEDURE UNIT"
  * Procedure LvlCalc;
  * var
-   Temp: Real;
  * Begin
  *   Case Exp of
  *     0..100: Level:=1;
  *     101..300: Level:=2;
40  *     301..500: Level:=3;
  *     501..1000: Level:=4;
  *     1001..3000: Level:=5;
  *     3001..5000: Level:=6;
  *     5001..10000: Level:=7;
-   *     10001..30000: Level:=8;
  *     30001..50000: Level:=9;
  *     50001..100000: Level:=10;
  *   end;
  *   FrmMainUI.LblCharLvl.Caption:= 'LVL: '+inttostr(level)
50  End;
  *

```

Exp = Experience | In this case statement BETWEEN 0 & 100 Experience = Level 1

As an algorithm this can be written as:

```

Case number of
  1: < Statement 1>
  2: <Statement 2>
  3: <Statement 3>
  Else <statement4>
End

```

Delphi Chapter 5:

LOOPS:

On the next tutorial we will cover what loop is and how to use them, as well as what type of loops there are. Please download the next tutorial at: [Click Here!](#)