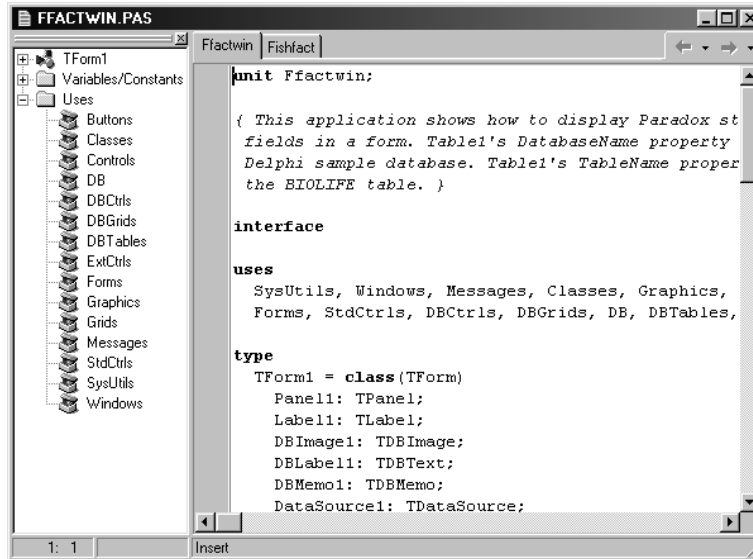


## Main Structure of a Delphi Application

### FORM SOURCE FILE (pas extension)



Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Main Structure of a Delphi Application

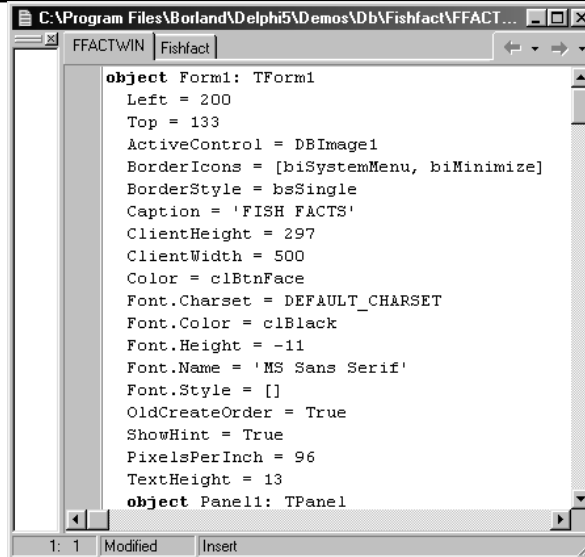
### APPEARANCE OF FORM OBJECTS



Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Main Structure of a Delphi Application

### FORM OBJECTS (DESIGN) FILE (dfm extension)



```
object Form1: TForm1
  Left = 200
  Top = 133
  ActiveControl = DBImage1
  BorderIcons = [biSystemMenu, biMinimize]
  BorderStyle = bsSingle
  Caption = 'FISH FACTS'
  ClientHeight = 297
  ClientWidth = 500
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clBlack
  Font.Height = -11
  Font.Name = 'MS Sans Serif'
  Font.Style = []
  OldCreateOrder = True
  ShowHint = True
  PixelsPerInch = 96
  TextHeight = 13
object Panell1: TPanel
```

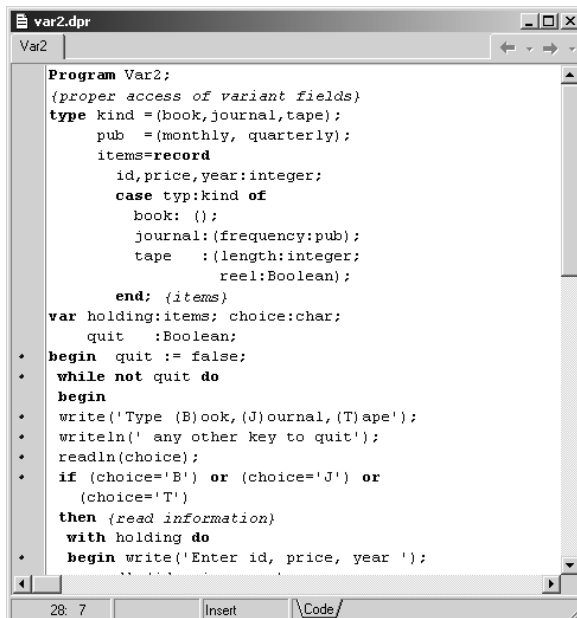
Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Main Structure of a Delphi Application

### Example: var2.dpr

This example is one single project file and does not use any form.

(Therefore, There is no *Uses* statement)



```
Program Var2;
{proper access of variant fields}
type kind = (book, journal, tape);
pub = (monthly, quarterly);
items = record
  id, price, year: integer;
  case typ: kind of
    book: ();
    journal: (frequency: pub);
    tape : (length: integer;
           reel: Boolean);
  end; {items}
var holding: items; choice: char;
quit : Boolean;
begin quit := false;
while not quit do
begin
write('Type (B)ook, (J)ournal, (T)ape');
writeln(' any other key to quit');
readln(choice);
if (choice='B') or (choice='J') or
(choice='T')
then {read information}
with holding do
begin write('Enter id, price, year ');
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

In this part, Delphi Integrated Development Environment is described.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

### Delphi Integrated Development Environment

Project Run Component Databases

- Add to Project... Shift+F11
- Remove from Project...
- Import Type Library...
- Add to Repository...
- View Source
- Languages
- Add New Project...
- Add Existing Project...
- Compile Var2 Ctrl+F9
- Build Var2
- Syntax check Var2
- Information for [none]
- Compile All Projects
- Build All Projects
- Web Deployment Options...
- Web Deploy
- Options... Shift+Ctrl+F11

**Check 'console application' choice to use write/read statements in Pascal language**

**Project Options**

Directories/Conditionals | Version Info | Packages

Forms | Application | Compiler | Linker

Map file

- Off
- Segments
- Publics
- Detailed

EXE and DLL options

- Generate console application
- Include IDE debug info
- Include remote debug symbols

Description

EXE Description:

Default

Linker output

- Generate DLLs
- Generate C object files
- Generate C++ object files
- Include namespaces
- Export all symbols

Memory sizes

Min stack size: \$00004000

Max stack size: \$00100000

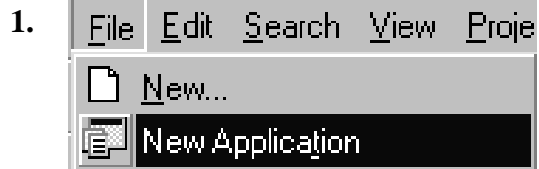
Image base: \$00400000

OK Cancel Help

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### First Sample



**Choose New Application**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

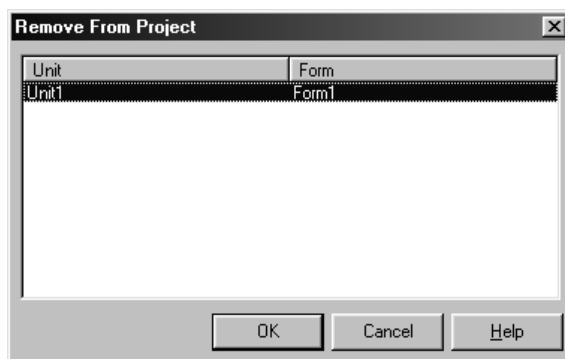
## Delphi Integrated Development Environment

### First Sample



*A Default form named as Form1 (source name is Unit1) is constructed.*

**Remove this form from the project.**



Assoc.Prof.Dr.B.G.Çetiner ? 2000

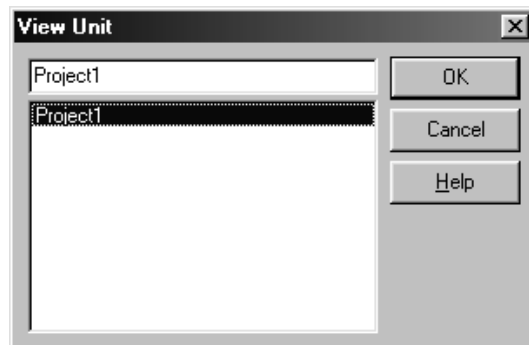
## Delphi Integrated Development Environment

### First Sample

3.



For viewing the project code press the view unit button and choose Project1 file from the dialog box as below.

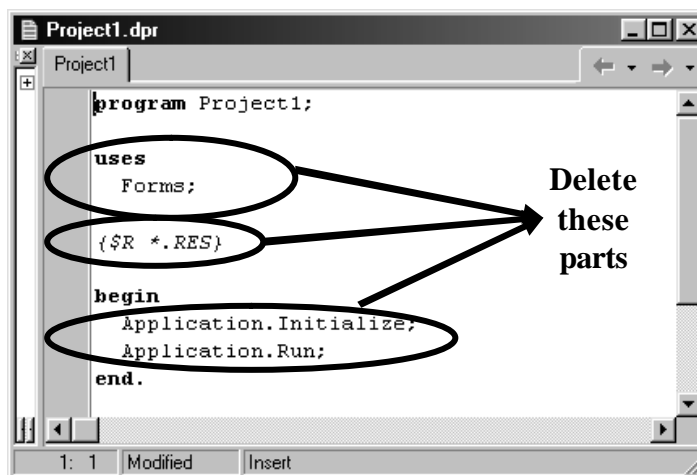


Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### First Sample

3.

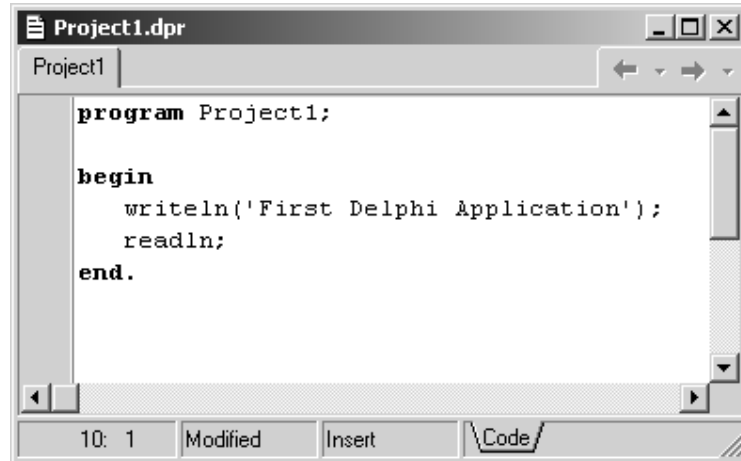


Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### First Sample

4.



```
program Project1;
begin
  writeln('First Delphi Application');
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

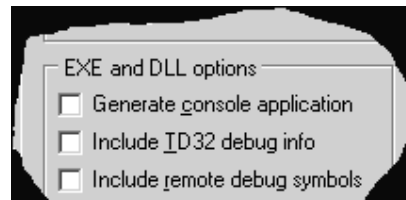
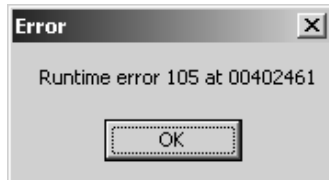
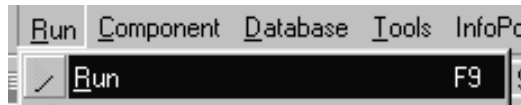
## Delphi Integrated Development Environment

### First Sample

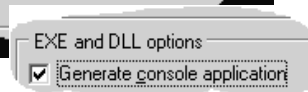
5. Compile and Run the program



or



If runtime error occurs then  
Go to Project/Options and check  
The *Generate console application* option



Assoc.Prof.Dr.B.G.Çetiner ? 2000

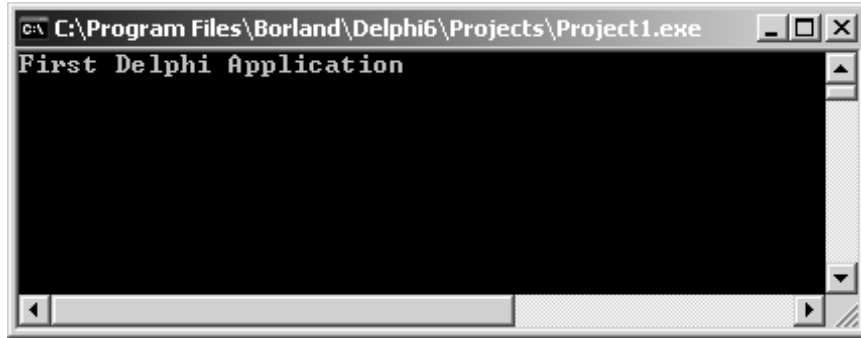


## Delphi Integrated Development Environment

### First Sample

### Screen Output

6.



Console Application is run in a DOS screen console

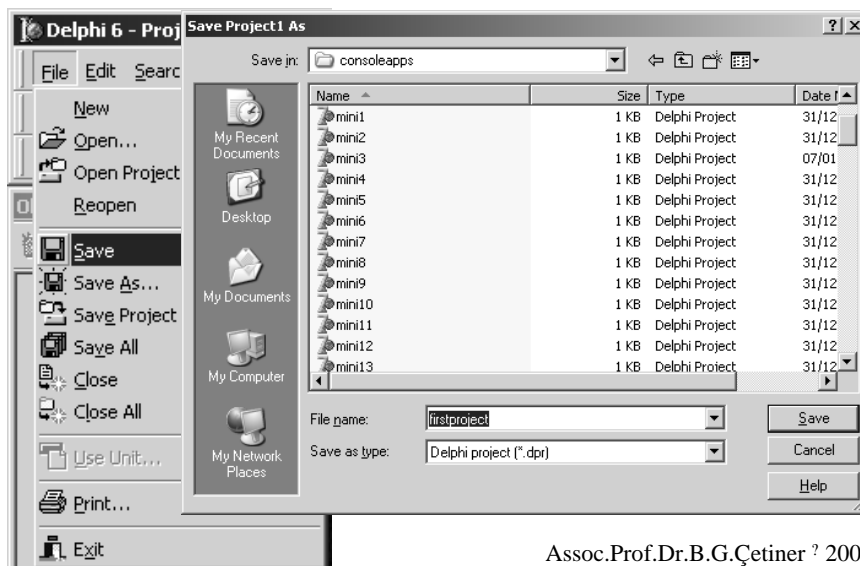
Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### First Sample

### For saving the Project

6.



Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### First Sample

Project was saved as 'firstproject'.



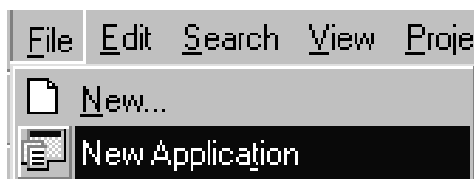
```
program firstproject;  
begin  
    writeln('First Delphi Application');  
    readln;  
end.
```

Writeln and readln statements are not normally used in visual applications. They are basic input-output statements under standard pascal.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### An application including a simple form



Choose new application and run it directly.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

An application including a simple form

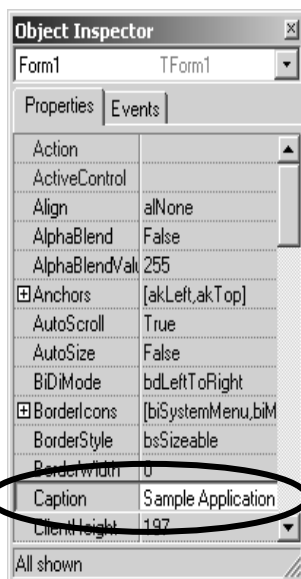


New application is a windows program with a single form.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

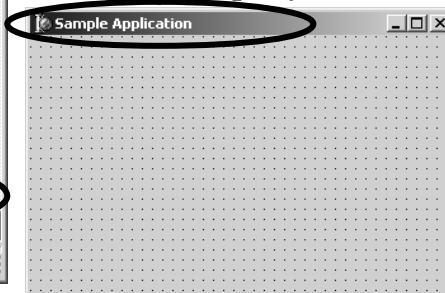
## Delphi Integrated Development Environment

### Object Inspector Window



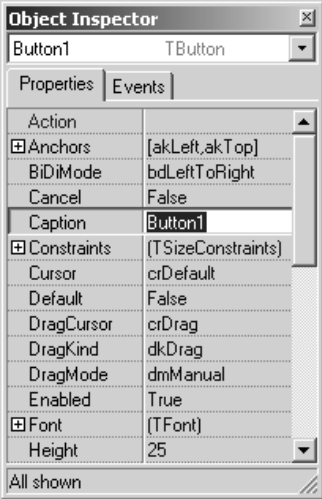
You can change the properties of selected objects in *Properties* page of Object Inspector Window.

Example: You select Form object and then, you can change properties of Form1 object. Enter the new value 'Sample Application' for caption property.



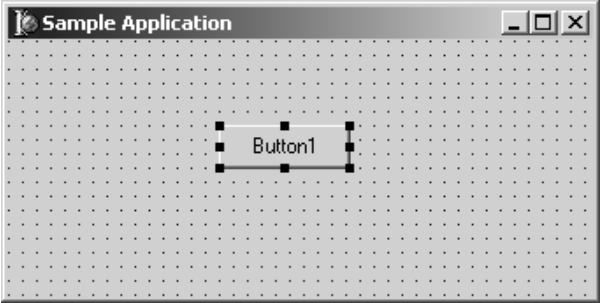
Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Delphi Integrated Development Environment** **Object Inspector Window**



**Object Inspector**  
 Button1 TButton  
 Properties Events  
 Action  
 Anchors [akLeft,akTop]  
 BiDiMode bdLeftToRight  
 Cancel False  
 Caption **Button1**  
 Constraints (TSizeConstraints)  
 Cursor crDefault  
 Default False  
 DragCursor crDrag  
 DragKind dkDrag  
 DragMode dmManual  
 Enabled True  
 Font (TFont)  
 Height 25  
 All shown

Now Choose a TButton object from Components Palette and place on the form. Now under properties Page, you can change the properties of TButton object.



**Sample Application**

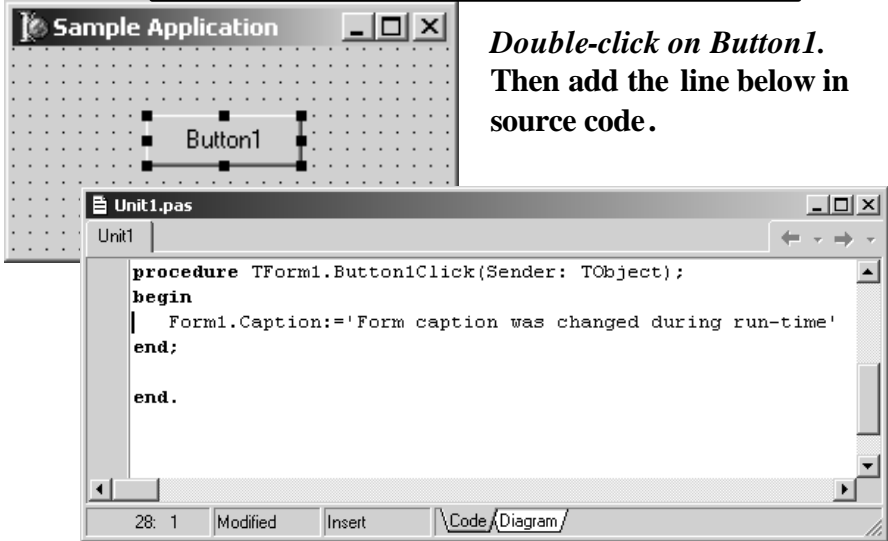
Button1

These changes are performed in *Design Time*.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Delphi Integrated Development Environment**

**Changing object properties during run-time**



**Sample Application**

Button1

**Unit1.pas**  
 Unit1  

```

procedure TForm1.Button1Click(Sender: TObject);
begin
  Form1.Caption:='Form caption was changed during run-time'
end;

end.

```

 28: 1 Modified Insert \Code/Diagram/

*Double-click on Button1.*  
 Then add the line below in source code.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Delphi Integrated Development Environment

### Changing object properties during run-time



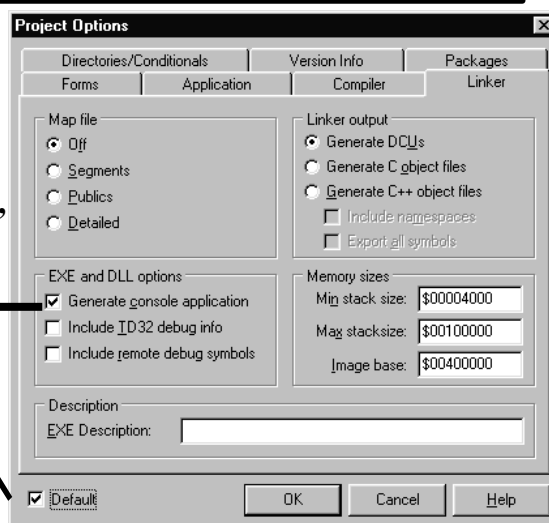
Press *Button1* button to change the Form caption when program is running.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

*Examples regarding Basics of Language are in CONSOLEAPPS folder.*

*For Basics of Language examples, choose these two checkboxes.*



Assoc.Prof.Dr.B.G.Çetiner ? 2000

<b>Basics of Language</b>	<b>Variables and Data Types</b>
<i>Sample program mini2.dpr in consoleapps folder</i>	
<p><b>Note:</b> Samples for Basics of Language are given as <i>console</i> applications under <i>consoleapps</i> directory</p> <p style="text-align: right;">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>	

<b>Basics of Language</b>	<b>Variables and Data Types</b>
<b>Pascal Program Structure</b>	
<p><b>Declaration Block:</b> Consists of <i>Variable</i>, <i>procedure</i> and <i>function</i> definitions.</p>	<pre> Program ProgramName; Const ..... Type ..... Var ..... Procedure ..... ..... Function ..... ..... begin ..... ..... ..... end. </pre>
<p style="text-align: center;">Declaration Block</p> <p style="text-align: center;">Execution Block</p>	<p style="text-align: right;">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

### String Data Type

Type	Maximum Length	Required Memory
ShortString	255 characters	2-256 bytes
AnsiString	$\sim 2^{31}$ characters	4 bayt-2GB
WideString	$\sim 2^{30}$ characters	4 bayt-2GB

If you define a variable as *String*, it is treated as *ansistring*.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini3.dpr* in *consoleapps* folder

```

Program mini3;
{Variable value can change}
var name : string;
begin
  write('What is your father''s name? ');
  readln(name);
  writeln('Your Father is : ',name);
  write('What is your mother''s name? ');
  readln(name);
  writeln('Your Mother is : ',name);
  readln;
end.

```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini4.dpr* in *consoleapps* folder

```

mini4.dpr
mini4
Program mini4;
{using two variables}
var father,mother : string;
begin
  write('Name of your father? ');
  readln(father);
  write('Name of your mother? ');
  readln(mother);
  write('Father is ',father,
        ' Mother is ',mother);
  readln
end.
1: 1      Insert      \Code/
    
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

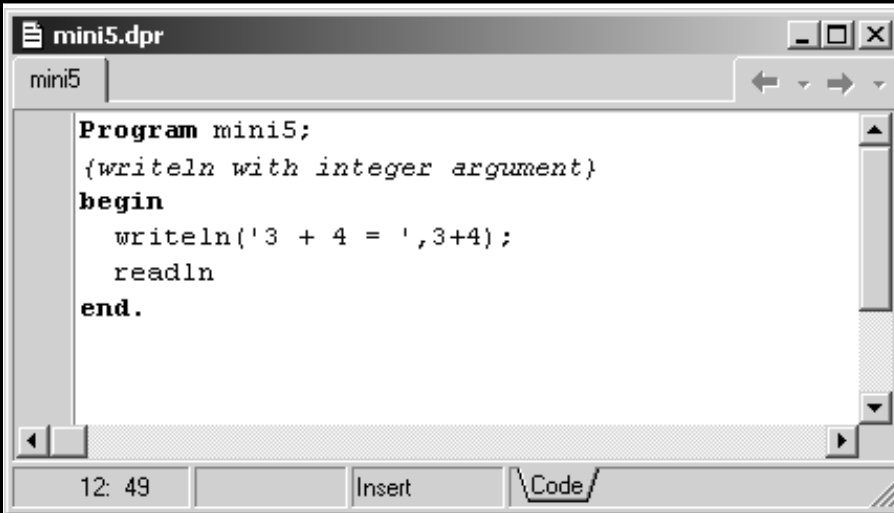
Integer Data Types

Type	Range	Memory Req.
Integer	-2147483648..2147483647	4
Cardinal	0..4294967295	4
Shortint	-128..127	1
Smallint	-32768..32767	2
Longint	-2147483648..2147483647	4
Int64	-2 <sup>63</sup> ..2 <sup>63</sup> -1	8
Byte	0..255	1
Word	0..65535	2
Longword	0..4294967295	4

Assoc.Prof.Dr.B.G.Çetiner ? 2000



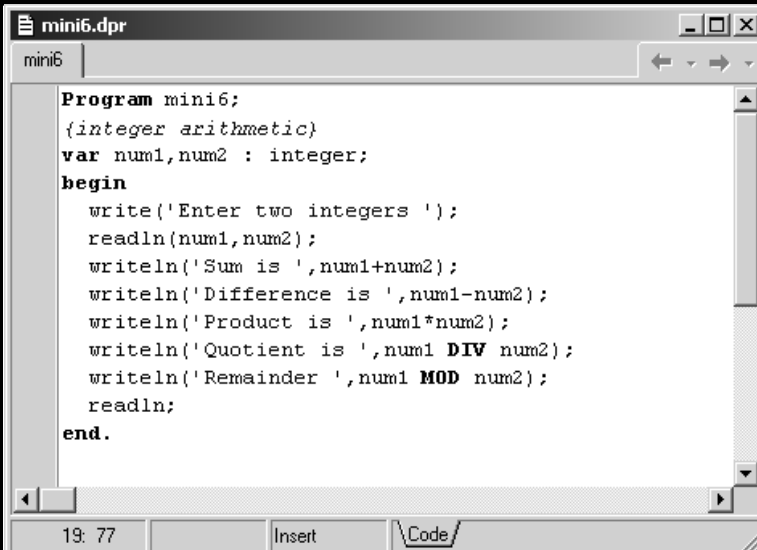
Sample program *mini5.dpr* in *consoleapps* folder



```
Program mini5;
{writeln with integer argument}
begin
  writeln('3 + 4 = ',3+4);
  readln
end.
```

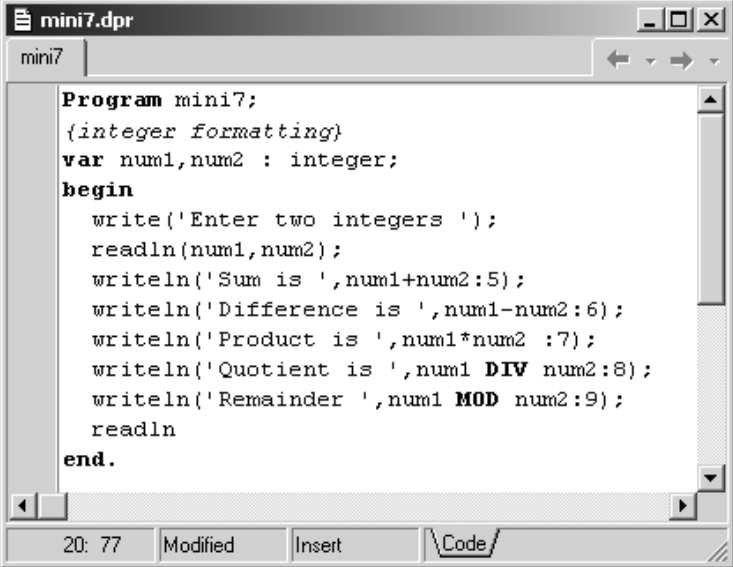
Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini6.dpr* in *consoleapps* folder



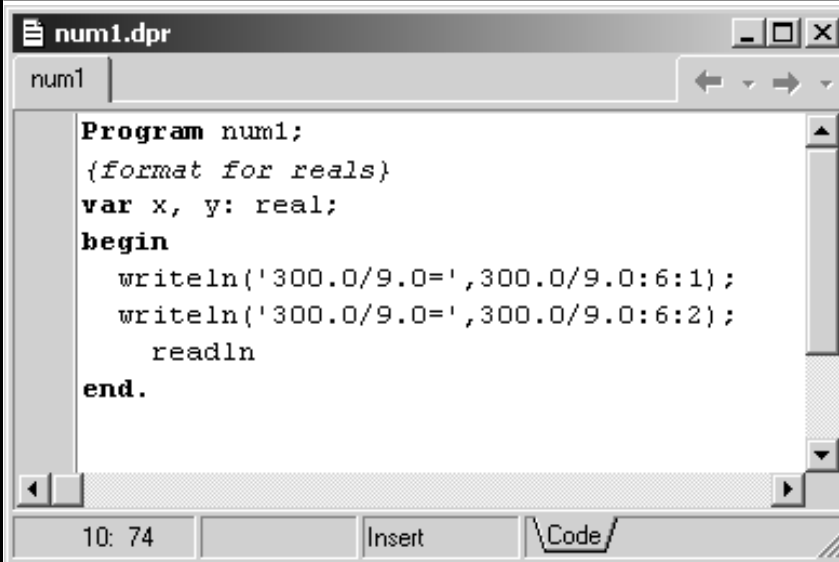
```
Program mini6;
{integer arithmetic}
var num1,num2 : integer;
begin
  write('Enter two integers ');
  readln(num1,num2);
  writeln('Sum is ',num1+num2);
  writeln('Difference is ',num1-num2);
  writeln('Product is ',num1*num2);
  writeln('Quotient is ',num1 DIV num2);
  writeln('Remainder ',num1 MOD num2);
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

<b>Basics of Language</b>	<b>Variables and Data Types</b>
<i>Sample program mini7.dpr in consoleapps folder</i>	
 <pre> Program mini7; (integer formatting) var num1,num2 : integer; begin   write('Enter two integers ');   readln(num1,num2);   writeln('Sum is ',num1+num2:5);   writeln('Difference is ',num1-num2:6);   writeln('Product is ',num1*num2 :7);   writeln('Quotient is ',num1 DIV num2:8);   writeln('Remainder ',num1 MOD num2:9);   readln end. </pre>	
Assoc.Prof.Dr.B.G.Çetiner ? 2000	

<b>Basics of Language</b>	<b>Variables and Data Types</b>																																
<b>Real Data Types</b>																																	
<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 15%;">Type</th> <th style="width: 35%;">Range</th> <th style="width: 20%;">Number of Stored Digits</th> <th style="width: 30%;">Memory Req. (bytes)</th> </tr> </thead> <tbody> <tr> <td>Real48</td> <td><math>2.9 \times 10^{-39} \dots 1.7 \times 10^{38}</math></td> <td>11-12</td> <td>6</td> </tr> <tr> <td>Single</td> <td><math>1.5 \times 10^{-45} \dots 3.4 \times 10^{38}</math></td> <td>7-8</td> <td>4</td> </tr> <tr> <td>Double</td> <td><math>5.0 \times 10^{-324} \dots 1.7 \times 10^{308}</math></td> <td>15-16</td> <td>8</td> </tr> <tr> <td>Extended</td> <td><math>3.6 \times 10^{-4951} \dots 1.1 \times 10^{4932}</math></td> <td>19-20</td> <td>10</td> </tr> <tr> <td>Comp</td> <td><math>-2^{63} + 1 \dots 2^{63} - 1</math></td> <td>19-20</td> <td>8</td> </tr> <tr> <td>Currency</td> <td>-922337203685477.5808.. 922337203685477.5807</td> <td>19-20</td> <td>8</td> </tr> <tr> <td>Real</td> <td><math>5.0 \times 10^{-324} \dots 1.7 \times 10^{308}</math></td> <td>15-16</td> <td>8</td> </tr> </tbody> </table>		Type	Range	Number of Stored Digits	Memory Req. (bytes)	Real48	$2.9 \times 10^{-39} \dots 1.7 \times 10^{38}$	11-12	6	Single	$1.5 \times 10^{-45} \dots 3.4 \times 10^{38}$	7-8	4	Double	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$	15-16	8	Extended	$3.6 \times 10^{-4951} \dots 1.1 \times 10^{4932}$	19-20	10	Comp	$-2^{63} + 1 \dots 2^{63} - 1$	19-20	8	Currency	-922337203685477.5808.. 922337203685477.5807	19-20	8	Real	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$	15-16	8
Type	Range	Number of Stored Digits	Memory Req. (bytes)																														
Real48	$2.9 \times 10^{-39} \dots 1.7 \times 10^{38}$	11-12	6																														
Single	$1.5 \times 10^{-45} \dots 3.4 \times 10^{38}$	7-8	4																														
Double	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$	15-16	8																														
Extended	$3.6 \times 10^{-4951} \dots 1.1 \times 10^{4932}$	19-20	10																														
Comp	$-2^{63} + 1 \dots 2^{63} - 1$	19-20	8																														
Currency	-922337203685477.5808.. 922337203685477.5807	19-20	8																														
Real	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$	15-16	8																														
Assoc.Prof.Dr.B.G.Çetiner ? 2000																																	

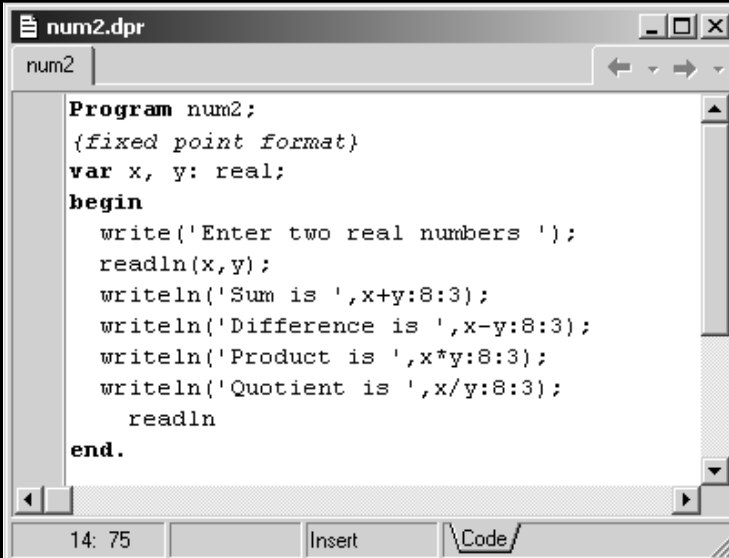
Sample program *num1.dpr* in *consoleapps* folder



```
Program num1;
{format for reals}
var x, y: real;
begin
  writeln('300.0/9.0=',300.0/9.0:6:1);
  writeln('300.0/9.0=',300.0/9.0:6:2);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

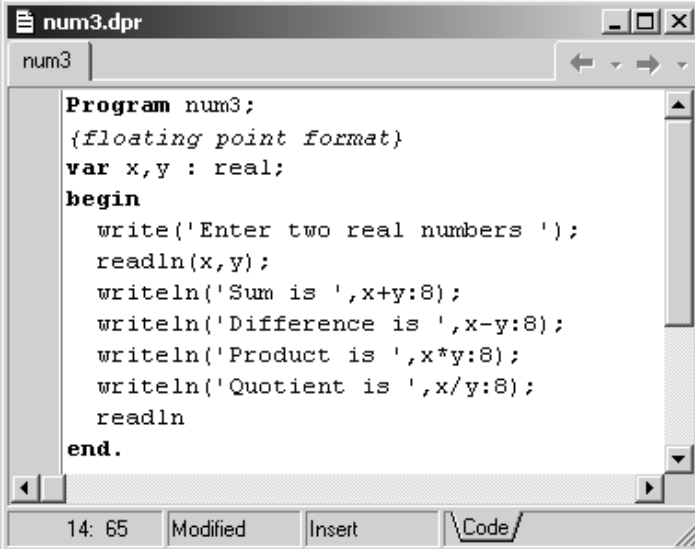
Sample program *num2.dpr* in *consoleapps* folder



```
Program num2;
{fixed point format}
var x, y: real;
begin
  write('Enter two real numbers ');
  readln(x,y);
  writeln('Sum is ',x+y:8:3);
  writeln('Difference is ',x-y:8:3);
  writeln('Product is ',x*y:8:3);
  writeln('Quotient is ',x/y:8:3);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

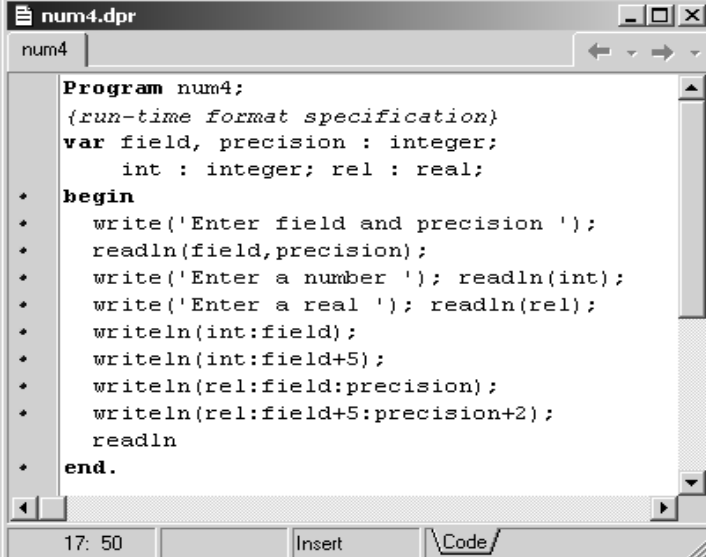
Sample program *num3.dpr* in consoleapps folder



```
num3
Program num3;
{floating point format}
var x,y : real;
begin
  write('Enter two real numbers ');
  readln(x,y);
  writeln('Sum is ',x+y:8);
  writeln('Difference is ',x-y:8);
  writeln('Product is ',x*y:8);
  writeln('Quotient is ',x/y:8);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

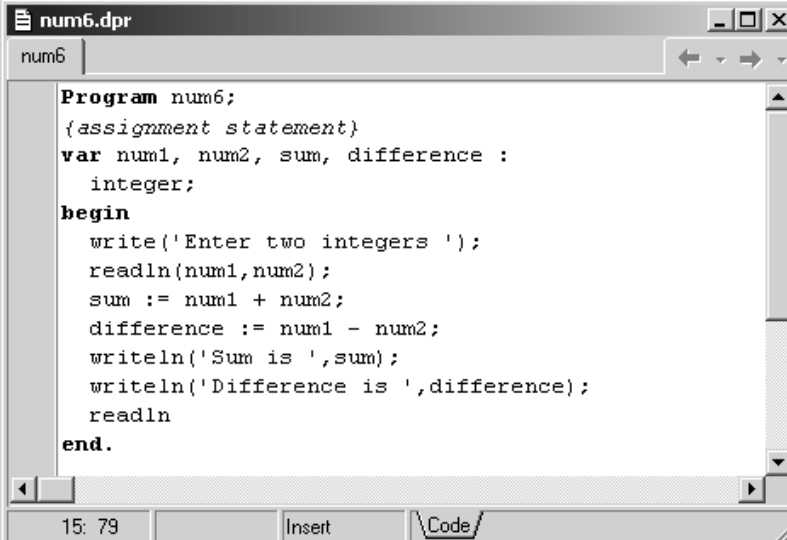
Sample program *num4.dpr* in consoleapps folder



```
num4
Program num4;
{run-time format specification}
var field, precision : integer;
    int : integer; rel : real;
begin
  write('Enter field and precision ');
  readln(field,precision);
  write('Enter a number '); readln(int);
  write('Enter a real '); readln(rel);
  writeln(int:field);
  writeln(int:field+5);
  writeln(rel:field:precision);
  writeln(rel:field+5:precision+2);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *num6.dpr* in *consoleapps* folder



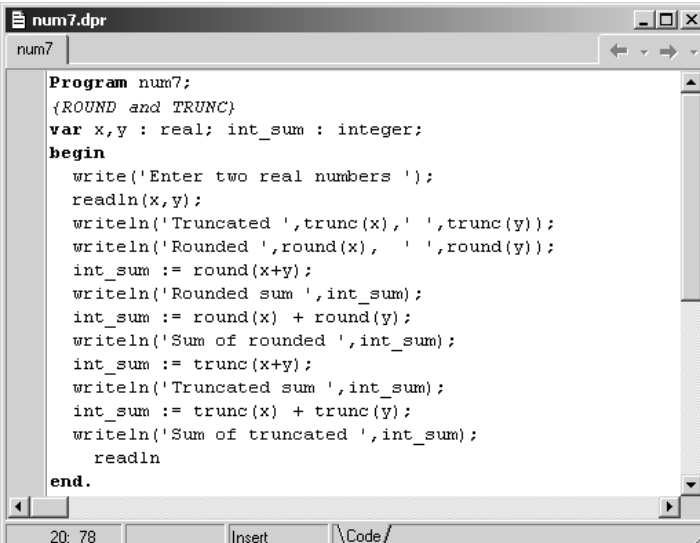
```

Program num6;
{assignment statement}
var num1, num2, sum, difference :
    integer;
begin
    write('Enter two integers ');
    readln(num1,num2);
    sum := num1 + num2;
    difference := num1 - num2;
    writeln('Sum is ',sum);
    writeln('Difference is ',difference);
    readln
end.

```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *num7.dpr* in *consoleapps* folder

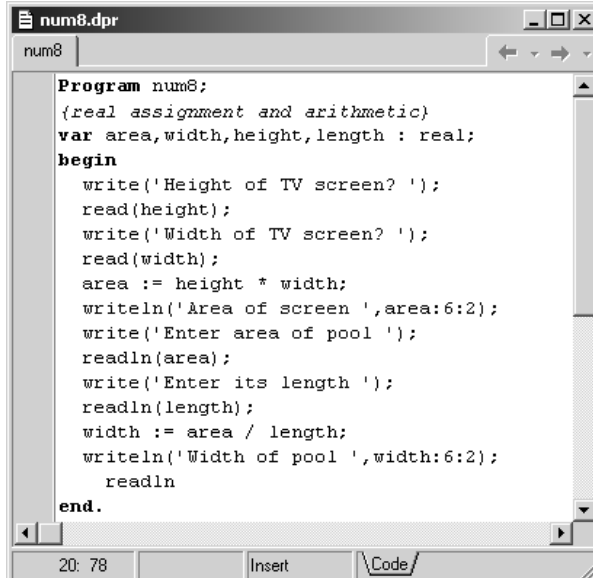


```

Program num7;
{ROUND and TRUNC}
var x,y : real; int_sum : integer;
begin
    write('Enter two real numbers ');
    readln(x,y);
    writeln('Truncated ',trunc(x),' ',trunc(y));
    writeln('Rounded ',round(x),' ',round(y));
    int_sum := round(x+y);
    writeln('Rounded sum ',int_sum);
    int_sum := round(x) + round(y);
    writeln('Sum of rounded ',int_sum);
    int_sum := trunc(x+y);
    writeln('Truncated sum ',int_sum);
    int_sum := trunc(x) + trunc(y);
    writeln('Sum of truncated ',int_sum);
    readln
end.

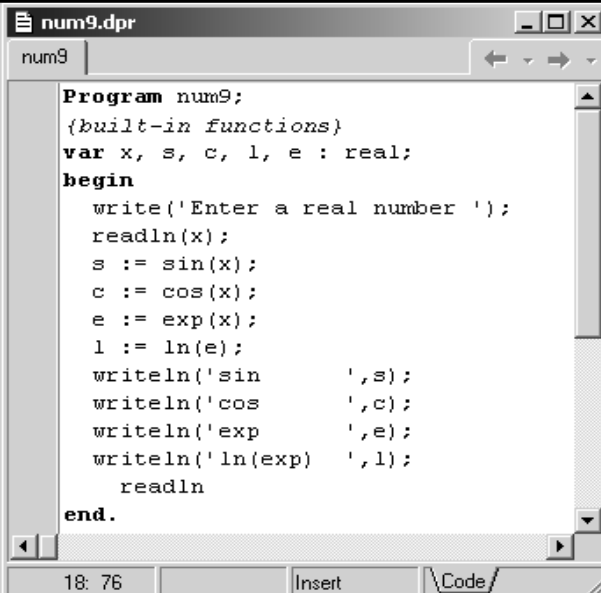
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program num8.dpr in consoleapps folder*

```
num8.dpr
num8
Program num8;
{real assignment and arithmetic}
var area,width,height,length : real;
begin
  write('Height of TV screen? ');
  read(height);
  write('Width of TV screen? ');
  read(width);
  area := height * width;
  writeln('Area of screen ',area:6:2);
  write('Enter area of pool ');
  readln(area);
  write('Enter its length ');
  readln(length);
  width := area / length;
  writeln('Width of pool ',width:6:2);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program num9.dpr in consoleapps folder*

```
num9.dpr
num9
Program num9;
{built-in functions}
var x, s, c, l, e : real;
begin
  write('Enter a real number ');
  readln(x);
  s := sin(x);
  c := cos(x);
  e := exp(x);
  l := ln(e);
  writeln('sin      ',s);
  writeln('cos      ',c);
  writeln('exp      ',e);
  writeln('ln(exp)  ',l);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini8.dpr* in consoleapps folder

```

Program mini8;
{integer and string variables}
var f_name,m_name : string;
    f_age,m_age : integer;
begin
    write('Father''s name? ');
    readln(f_name);
    write('Mother''s name? ');
    readln(m_name);
    write('Father''s and', ' mother''s age? ');
    readln(f_age,m_age);
    writeln('Father ',f_name,' is ', f_age,' years old');
    writeln('Mother ',m_name,' is ', m_age,' years old');
    readln
end.
    
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini9.dpr* in consoleapps folder

```

Program mini9;
{text files}
var f : text; x : integer;
begin
    writeln('Writing to file1');
    assign(f,'file1'); {step 1}
    rewrite(f); {step 2}
    {step 3 - writing}
    writeln(f,3+4,' ',17-25,' ',8,' ');
    writeln(f,-100);
    close(f); {step 4}
    {now use f for reading - step 1 done}
    writeln('Reading from file1');
    {step 2 - set file for reading}
    reset(f);
    {step 3 - reading}
    read(f,x); writeln(x);
    read(f,x); writeln(x);
    close(f); {step 4}
    readln
end.
    
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Variables and Data Types

```
mini10.dpr
mini10
Program mini10;
{two files}
var f1,f2 : text;
    fname : string;
    x : integer;
begin
  write('We are going to read from ',
        'file1 and write into another file');
  write('Enter name of second file ');
  readln(fname);
  assign(f1,'file1');
  assign(f2,fname);
  {note use of string variable above}
  reset(f1);
  rewrite(f2);
  read(f1,x); write(f2,x,' ');
  write('Enter a number '); readln(x);
  writeln(f2,x);
  read(f1,x); write(f2,x,' ');
  write('Enter a number '); readln(x);
  writeln(f2,x);
  close(f1); close(f2);
  readln
end.
```

*Sample program mini10.dpr  
in consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Variables and Data Types

*Sample program mini12.dpr in consoleapps folder*

```
mini12.dpr
mini12
Program mini12;
{declared string and integer constants}
const c1 = 32; c2 = 5; c3 = 9;
      prompt = 'Enter Fahrenheit ';
      answer = ' degrees Celsius';
var fahr, cels : integer;
begin
  write(prompt); readln(fahr);
  cels := ((fahr-c1) * c2) div c3;
  write(cels,answer);
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000



```
if { condition1 } then
  { statement1 }
else
  {statement2};
```

```
if ... { condition1 } then
begin
  if ... {condition2 } then
    ... { statement1 }
  else
    ... { statement2 }
end;
```

**Examples**

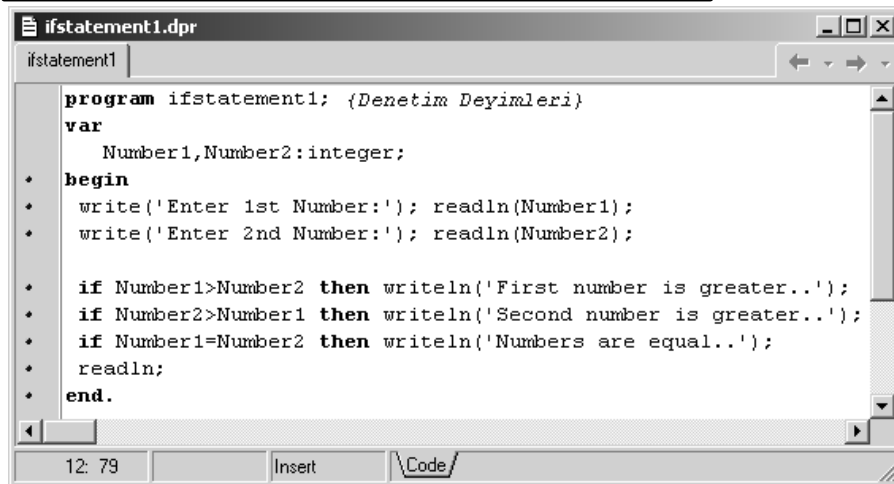
**If there are more than one statements, statements are put between begin-end words as a block**

```
if x>12 then y:=x*2;

if Salary>10000 then
begin
  Tax:=Oran1*SalaryBeforeTax;
  Net:=SalaryBeforeTax-Tax;
end;

if Delta<0 then
  message:='imaginary numbers'
else
begin
  x1:=(-b+Sqrt(Delta))/2*a;
  x2:=(-b-Sqrt(Delta))/2*a;
end;
```

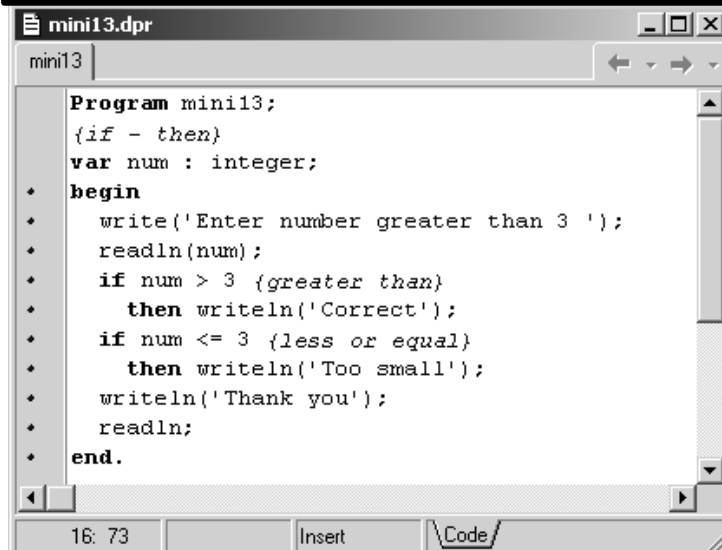
Sample program *ifstatement1.dpr* in *consoleapps* folder



```
ifstatement1.dpr
ifstatement1
program ifstatement1; {Denetim Deyimleri}
var
    Number1,Number2:integer;
• begin
• write('Enter 1st Number:'); readln(Number1);
• write('Enter 2nd Number:'); readln(Number2);
•
• if Number1>Number2 then writeln('First number is greater..');
• if Number2>Number1 then writeln('Second number is greater..');
• if Number1=Number2 then writeln('Numbers are equal..');
• readln;
• end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

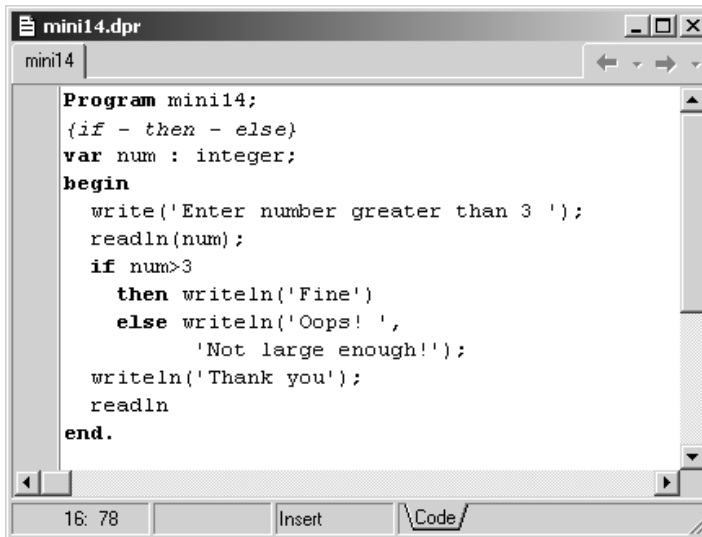
Sample program *mini13.dpr* in *consoleapps* folder



```
mini13.dpr
mini13
Program mini13;
{if - then}
var num : integer;
• begin
• write('Enter number greater than 3 ');
• readln(num);
• if num > 3 {greater than}
• then writeln('Correct');
• if num <= 3 {less or equal}
• then writeln('Too small');
• writeln('Thank you');
• readln;
• end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

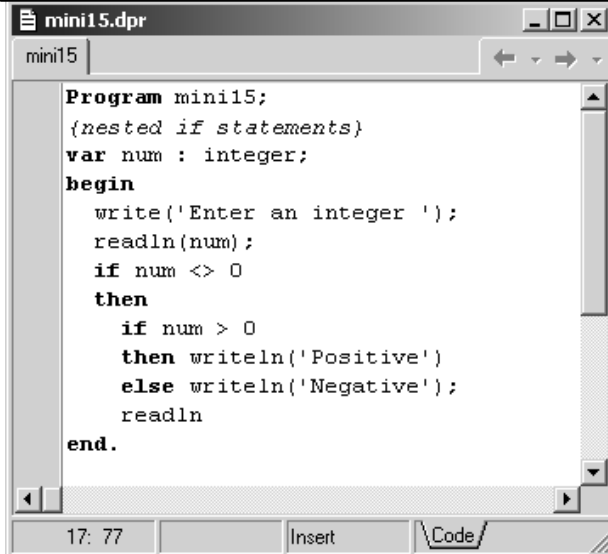
Sample program *mini14.dpr* in *consoleapps* folder



```
Program mini14;
{if - then - else}
var num : integer;
begin
  write('Enter number greater than 3 ');
  readln(num);
  if num>3
  then writeln('Fine')
  else writeln('Oops! ',
              'Not large enough!');
  writeln('Thank you!');
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

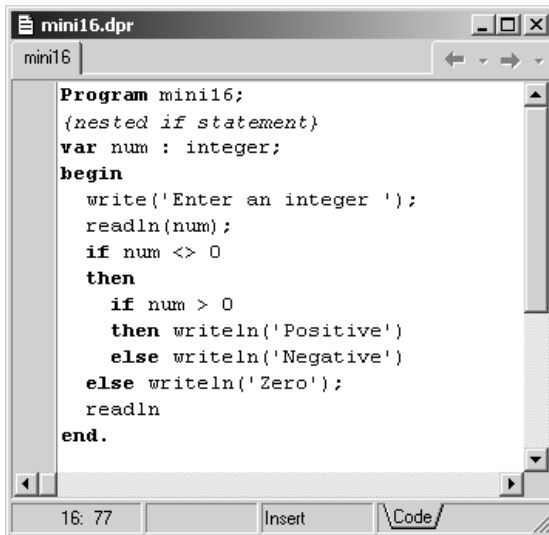
Sample program *mini15.dpr* in *consoleapps* folder



```
Program mini15;
{nested if statements}
var num : integer;
begin
  write('Enter an integer ');
  readln(num);
  if num <> 0
  then
    if num > 0
    then writeln('Positive')
    else writeln('Negative');
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

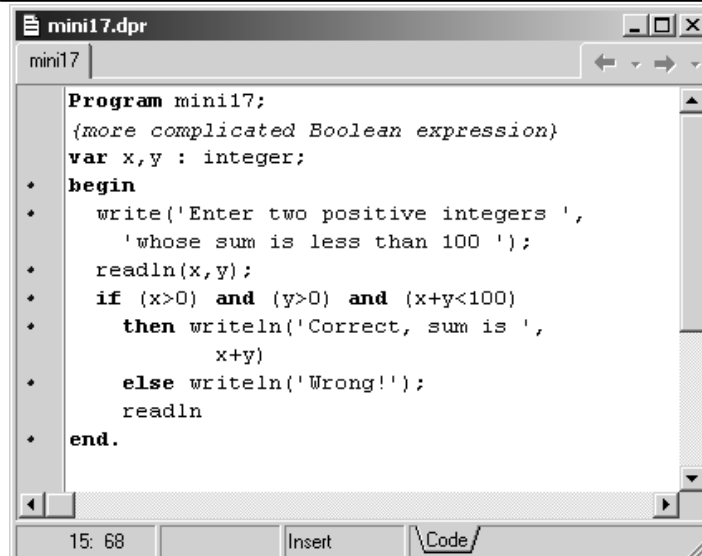
Sample program *mini16.dpr* in *consoleapps* folder



```
Program mini16;  
{nested if statement}  
var num : integer;  
begin  
  write('Enter an integer ');  
  readln(num);  
  if num <> 0  
  then  
    if num > 0  
    then writeln('Positive')  
    else writeln('Negative')  
  else writeln('Zero');  
  readln  
end.
```

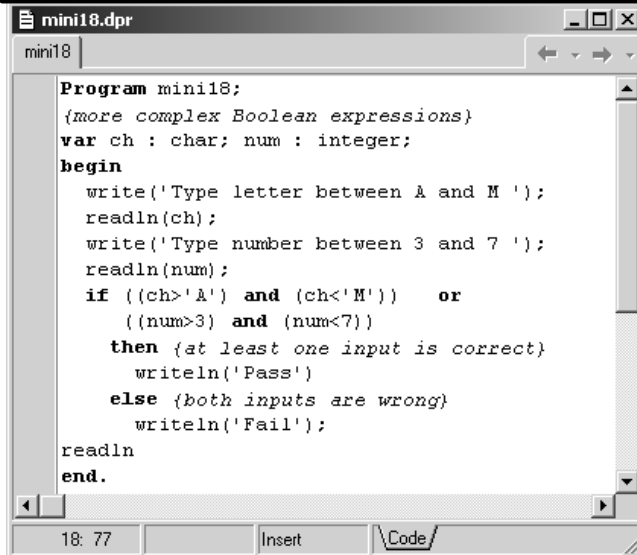
Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini17.dpr* in *consoleapps* folder



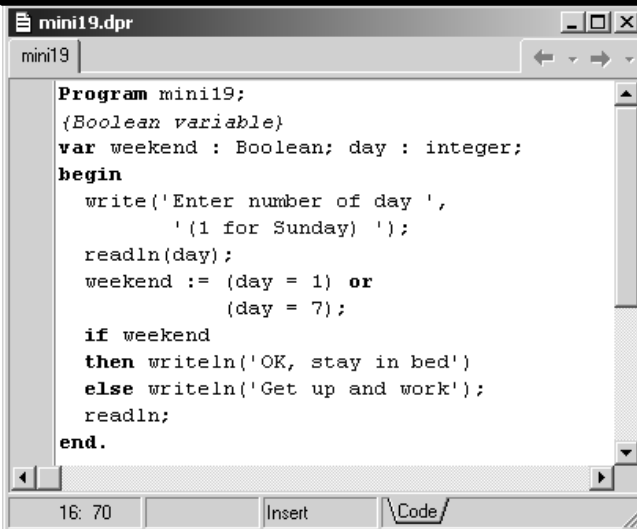
```
Program mini17;  
{more complicated Boolean expression}  
var x,y : integer;  
begin  
  write('Enter two positive integers ',  
        'whose sum is less than 100 ');  
  readln(x,y);  
  if (x>0) and (y>0) and (x+y<100)  
  then writeln('Correct, sum is ',  
              x+y)  
  else writeln('Wrong!');  
  readln  
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini18.dpr* in consoleapps folder**

```
Program mini18;
{more complex Boolean expressions}
var ch : char; num : integer;
begin
  write('Type letter between A and M ');
  readln(ch);
  write('Type number between 3 and 7 ');
  readln(num);
  if ((ch>'A') and (ch<'M')) or
     ((num>3) and (num<7))
  then {at least one input is correct}
    writeln('Pass')
  else {both inputs are wrong}
    writeln('Fail');
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini19.dpr* in consoleapps folder**

```
Program mini19;
{Boolean variable}
var weekend : Boolean; day : integer;
begin
  write('Enter number of day ',
        '(1 for Sunday) ');
  readln(day);
  weekend := (day = 1) or
            (day = 7);
  if weekend
  then writeln('OK, stay in bed')
  else writeln('Get up and work');
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini20.dpr* in *consoleapps* folder

```

Program mini20;
{Boolean variables in expressions}
var time : integer;
    weather : char; late : boolean;
begin
  write('What time is it? ');
  readln(time);
  late := (time<6) or (time>10);
  write('Is the weather good or bad ',
        '(g/b)? ');
  readln(weather);
  if (not late) and (weather='g')
  then writeln('Take a walk outside')
  else writeln('Work harder!');
  readln
end.
    
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

```

case Option of
  List1: statement1;
  ...
  List_n: statement_n,
else
  statement;
end
    
```

**If there are more than one statements, these statements are put between begin-end Words.**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## EXAMPLE

```

case I of
  1..5: Caption := 'Small';
  6..9: Caption := 'Big';
  0, 10..99: Caption := 'Invalid Value';
else
  Caption := '';
end;

```

Both are  
same

```

if I in [1..5] then
  Caption := 'Small'
else if I in [6..10] then
  Caption := 'Big'
else if (I = 0) or (I in [10..99]) then
  Caption := 'Invalid Value'
else
  Caption := '';

```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Examples

```

case Colour of
  Red: begin
    X := 1; y:=x*2;
  end;
  Green: X := 2;
  Blue: X := 3;
  Yellow, Orange, Black: X := 0;
end;

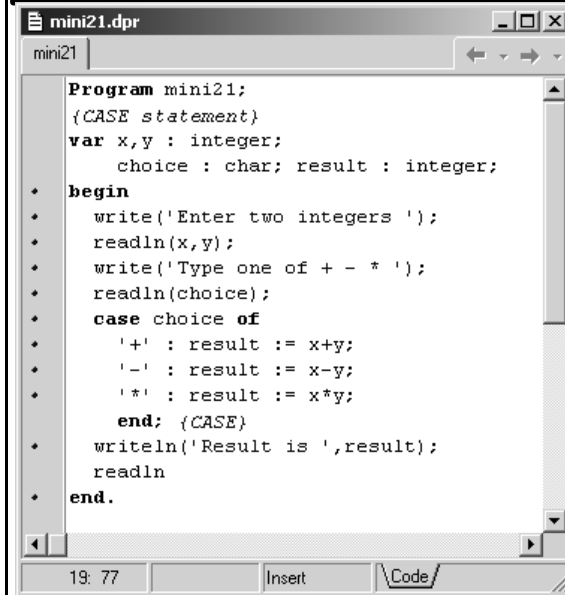
```

```

case Selection of
  OK: Form1.Close;
  Calculate: CalculateTotal(UnitCost, Quantity);
else
  Beep;
end;

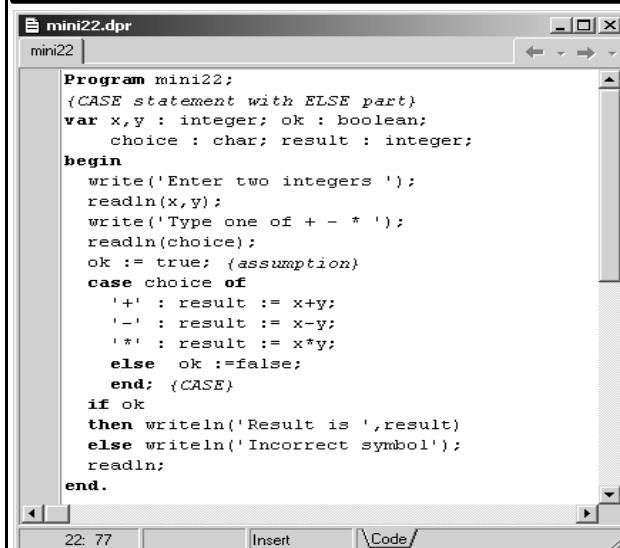
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program mini21.dpr in consoleapps folder*

```
Program mini21;
{CASE statement}
var x,y : integer;
    choice : char; result : integer;
begin
  write('Enter two integers ');
  readln(x,y);
  write('Type one of + - * ');
  readln(choice);
  case choice of
    '+' : result := x+y;
    '-' : result := x-y;
    '*' : result := x*y;
  end; {CASE}
  writeln('Result is ',result);
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program mini22.dpr in consoleapps folder*

```
Program mini22;
{CASE statement with ELSE part}
var x,y : integer; ok : boolean;
    choice : char; result : integer;
begin
  write('Enter two integers ');
  readln(x,y);
  write('Type one of + - * ');
  readln(choice);
  ok := true; {assumption}
  case choice of
    '+' : result := x+y;
    '-' : result := x-y;
    '*' : result := x*y;
    else ok :=false;
  end; {CASE}
  if ok
  then writeln('Result is ',result)
  else writeln('Incorrect symbol');
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000



*Sample program mini23.dpr in consoleapps folder*

```
Program mini23;
{CASE with ELSE clause}
var i : integer;
begin
  write('Enter integer between 1 and ',
        '10 ');
  readln(i);
  case i of
    1,3,5,7,9 : writeln('Odd number');
    2,4,6,8,10 : writeln('Even number');
  else begin writeln('Wrong');
            write('Not between 1 and 10');
          end;
  end; {CASE}
  readln;
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

```
repeat
    statement1;
    ...;
    statement_n;
until conditionStatement
```

**Statements between repeat-until are repeated until the condition is met.**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

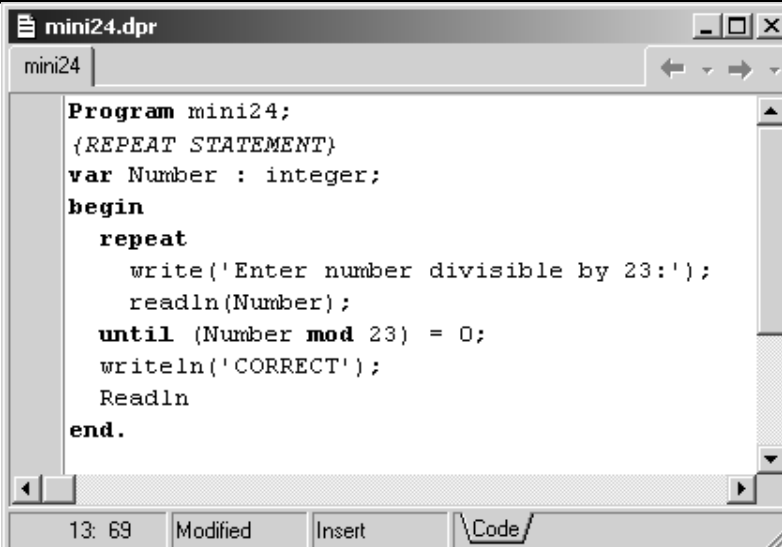
**EXAMPLE**

```
repeat  
  
    Write('Enter a value between 0..9 : ');  
    Readln(I);  
  
until (I >= 0) and (I <= 9);
```

Two statements between Repeat-until are repeated until user enters a value outside the range of 0..9

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini24.dpr* in consoleapps folder



```
Program mini24;  
{REPEAT STATEMENT}  
var Number : integer;  
begin  
    repeat  
        write('Enter number divisible by 23:');  
        readln(Number);  
    until (Number mod 23) = 0;  
    writeln('CORRECT');  
    Readln  
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Sample program *mini29.dpr* in *consoleapps* folder

```
mini29.dpr
mini29
Program mini29;
{getting the correct answer}
var answer : char; correct : Boolean;
begin
  repeat
    write('Answer y or n ');
    readln(answer);
    correct := (answer='y') or
              (answer='n');
    if not correct
      then writeln('Wrong answer!');
  until correct;
  writeln('Fine, thanks');
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

```
while condition do statement;
```

```
while condition do
begin
  statement1;
  .....
  statement_n;
end;
```

Statements are repeated as long as condition is valid.  
Condition is checked at the beginning.

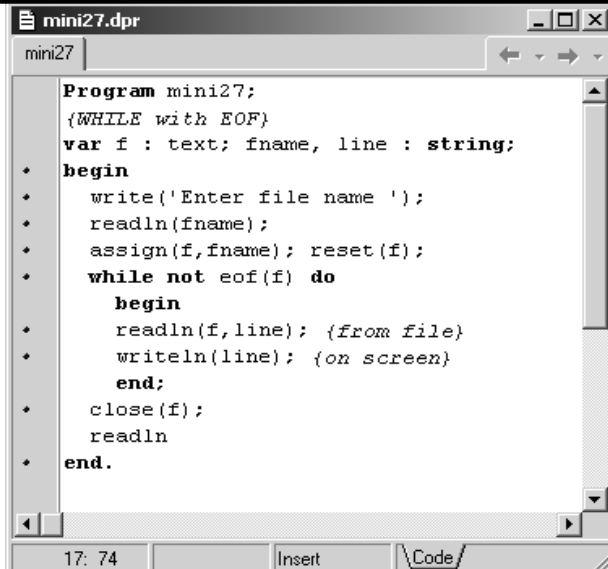
Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Examples

```
while Data[I] <> X do I := I + 1;
```

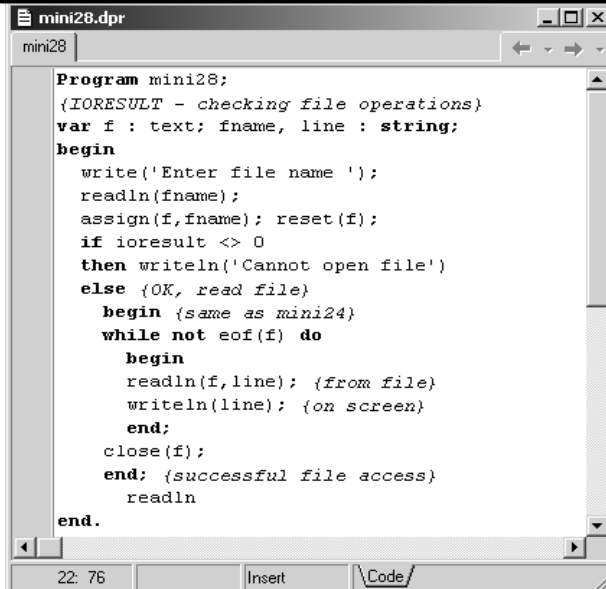
```
while I > 0 do  
begin  
  if Odd(I) then Z := Z * X;  
  I := I div 2;  
  X := Sqr(X);  
end;
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program mini27.dpr in consoleapps folder*

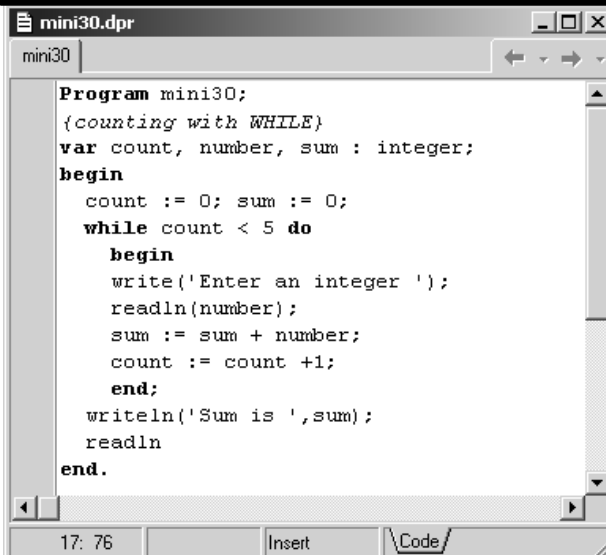
```
Program mini27;  
{WHILE with EOF}  
var f : text; fname, line : string;  
* begin  
*   write('Enter file name ');  
*   readln(fname);  
*   assign(f, fname); reset(f);  
*   while not eof(f) do  
*     begin  
*       readln(f, line); {from file}  
*       writeln(line); {on screen}  
*     end;  
*   close(f);  
*   readln  
* end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini28.dpr* in consoleapps folder**

```
Program mini28;
{IORESULT - checking file operations}
var f : text; fname, line : string;
begin
  write('Enter file name ');
  readln(fname);
  assign(f, fname); reset(f);
  if iorresult <> 0
  then writeln('Cannot open file')
  else {OK, read file}
  begin {same as mini24}
    while not eof(f) do
    begin
      readln(f, line); {from file}
      writeln(line); {on screen}
    end;
    close(f);
  end; {successful file access}
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini30.dpr* in consoleapps folder**

```
Program mini30;
{counting with WHILE}
var count, number, sum : integer;
begin
  count := 0; sum := 0;
  while count < 5 do
  begin
    write('Enter an integer ');
    readln(number);
    sum := sum + number;
    count := count + 1;
  end;
  writeln('Sum is ', sum);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

```
FOR variable := FirstValue TO LastValue DO statement;  
or  
FOR variable := FirstValue DOWNTO LastValue DO statement;
```

**Statement (or statements) are repeated starting from the first value until the last value.**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

### Examples

```
for I := 2 to 63 do  
  if Data[I] > Max then Max := Data[I];
```

```
for I := ListBox1.Items.Count - 1 downto 0 do  
  ListBox1.Items[I] := UpperCase(ListBox1.Items[I]);
```

```
for I := 1 to 10 do  
  for J := 1 to 10 do  
  begin  
    X := 0;  
    for K := 1 to 10 do  
      X := X + Mat1[I, K] * Mat2[K, J];  
    Mat[I, J] := X;  
  end;
```

```
for C := Red to Blue do Check(C);
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program mini31.dpr in consoleapps folder*

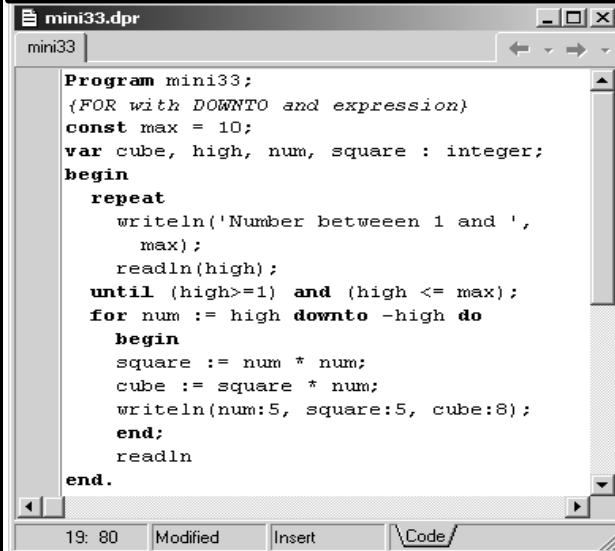
```
mini31.dpr
mini31
Program mini31;
{counting with FOR}
var sum, number, count : integer;
begin
  sum := 0;
  for count := 1 to 5 do
  begin
    write('Enter an integer ');
    readln(number);
    sum := sum + number;
  end;
  writeln('Sum is ',sum);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program mini32.dpr in consoleapps folder*

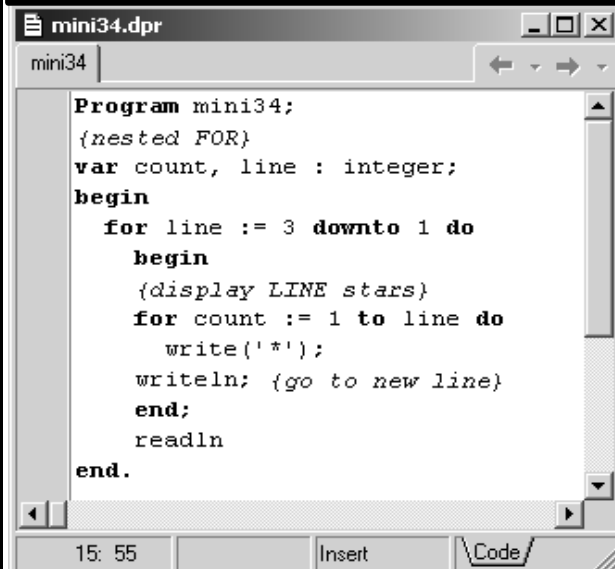
```
mini32.dpr
mini32
Program mini32;
{table of squares}
const max = 50;
var high, num : integer;
begin
  repeat
    writeln('Number between 1 ',
      'and ',max);
    readln(high);
  until (high>=1) and (high<=max);
  for num := 1 to high do
    writeln(num:5, num*num:10);
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini33.dpr* in consoleapps folder**

```
Program mini33;
{FOR with DOWNTO and expression}
const max = 10;
var cube, high, num, square : integer;
begin
  repeat
    writeln('Number between 1 and ',
      max);
    readln(high);
  until (high>=1) and (high <= max);
  for num := high downto -high do
  begin
    square := num * num;
    cube := square * num;
    writeln(num:5, square:5, cube:8);
  end;
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Sample program *mini34.dpr* in consoleapps folder**

```
Program mini34;
{nested FOR}
var count, line : integer;
begin
  for line := 3 downto 1 do
  begin
    {display LINE stars}
    for count := 1 to line do
      write('*');
    writeln; {go to new line}
  end;
  readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000



```
function Chr(X: Byte): Char;  
function UpCase(Ch: Char): Char;  
function Copy(S; Index, Count: Integer): string;  
procedure Delete(var S: string; Index, Count: Integer);  
procedure Insert(Source: string; var S: string; Index: Integer);  
function Concat(s1 [, s2, ..., sn]: string): string;  
function Length(S): Integer;  
function Pos(Substr: string; S: string): Integer;  
procedure Str(X [: Width [: Decimals ]]; var S);  
procedure Val(S; var V; var Code: Integer);
```

```
function SizeOf(X): Integer;  
function IsDelimiter(const Delimiters, S: string; Index: Integer): Boolean;  
function LowerCase(const S: string): string; (veya AnsiLowerCase)  
function UpperCase(const S: string): string; (veya AnsiUpperCase)  
function TrimLeft(const S: string): string;  
function TrimRight(const S: string): string;  
function Trim(const S: string): string;
```

Basics of Language

Functions for Character and String Manipulations

Sample program stringfunct.dpr in consoleapps folder

```

stringfunct.dpr
stringfunct
program stringfunct;
{$APPTYPE CONSOLE}
uses
  sysutils;

Var
  Line:string;
  i:integer; ch:char;
begin
  Line:='Istanbul to Jeddah';
  for i:=1 to length(Line) do
  begin
    ch:=Line[i];
    ch:=uppercase(ch);
    writeln(ch);
  end;
  readln;
end.
19: 57  Insert  \Code/

```

RESULT:



Assoc.Prof.Dr.B.G.Çetiner 2000

Basics of Language

Functions for Character and String Manipulations

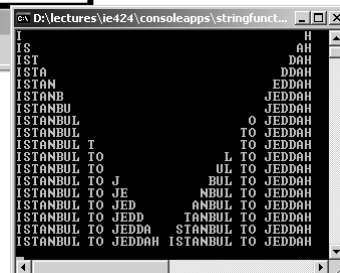
Sample program stringfunct2.dpr in consoleapps folder

RESULT:

```

stringfunct2.dpr
stringfunct2
program stringfunct2;
uses
  sysutils;
Var
  Line,TempLine1,Blanks,TempLine2:string;
  i:integer;
begin
  Line:='Istanbul to Jeddah';
  for i:=1 to length(Line) do
  begin
    TempLine1:=UpperCase(copy(Line, 1, i));
    TempLine2:=UpperCase(copy(Line, length(Line)-i+1, i));
    Blanks:=StringOfChar(' ', length(Line)*2-i+1-length(TempLine1));
    writeln(TempLine1,Blanks,TempLine2);
  end;
  readln;
end.
17: 73  Insert  \Code/

```



Assoc.Prof.Dr.B.G.Çetiner 2000

**Question:** Write a program which searches each line in a text file for a specific word or character given by the user and deletes the whole line if the specific word is matched the word in the searched line.

**Example;** The Word being searched: a

**The file being searched**

**Question:** Write a program which searches each line in a text file for a specific word or character given by the user and deletes the whole line if the specific word is matched the word in the searched line.

**The resulting file**

deletes the whole line if the specific word is line.

**All lines containing "a" have been deleted in the file.**

**Question:** Given a word by the user, write a computer program Which takes this word as input and counts the numbers, alphabetic and other chars in the input word and writes the frequencies for each.

**Example;** Word: 'Anka23[]andF2\''

**Frequency of alphabetic characters =8**  
**Frequency of Numbers =3**  
**Other Characters =4**

## Basics of Language

**New Data Types can be constructed via the use  
of TYPE and Record**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

**The simplest usage**

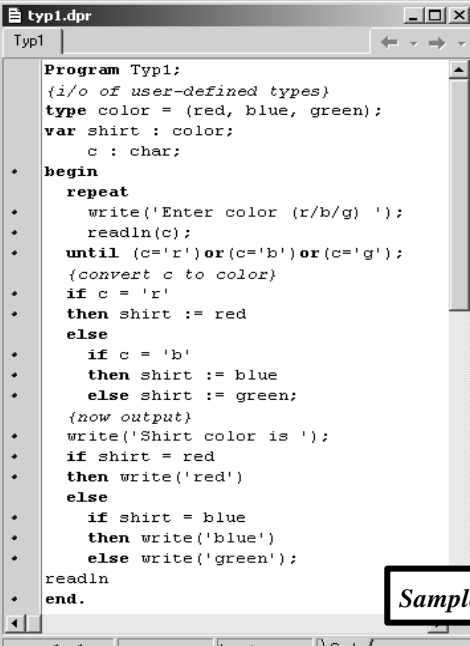
```
type TValue = Real;
```

```
var
```

```
  X: Real;
```

```
  Y: TValue;
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Basics of Language	User Defined Data Types
 <pre> Program Typ1; {i/o of user-defined types} type color = (red, blue, green); var shirt : color;     c : char; begin   repeat     write('Enter color (r/b/g) ');     readln(c);   until (c='r') or (c='b') or (c='g');   {convert c to color}   if c = 'r'   then shirt := red   else     if c = 'b'     then shirt := blue     else shirt := green;   {now output}   write('Shirt color is ');   if shirt = red   then write('red')   else     if shirt = blue     then write('blue')     else write('green');   readln; end. </pre>	<pre> type recordTypeName = record    fieldList1: type1;   ...   fieldListn: typen; end; Example type   TDateRec = record     Year: Integer;     Month: (Jan, Feb, Mar, Apr, May, Jun,             Jul, Aug, Sep, Oct, Nov, Dec);     Day: 1..31;   end; Define as Variable   var Record1, Record2: TDateRec; </pre> <p><i>Sample program typ1.dpr in consoleapps folder</i></p> <p>Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

Basics of Language	User Defined Data Types
<p>Example</p> <pre> type    TDateRec = record     Year: Integer;     Month: (Jan, Feb, Mar, Apr, May, Jun,             Jul, Aug, Sep, Oct, Nov, Dec);     Day: 1..31;   end; Define as Variable   var Record1, Record2: TDateRec; Usage   Record1.Year := 1904; OR  with Record1 do   Record1.Month := Jun;    begin   Record1.Day := 16;       Year := 1904;                            Month := Jun;                            Day := 16;                            end; </pre> <p>Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>	

## Basics of Language

## User Defined Data Types

Example

type

```
TEmployee = record  
  FirstName, LastName : string[40];  
  BirthDate: TDate;  
  case Salaried: Boolean of  
    True: (AnnualSalary: Currency);  
    False: (HourlyWage: Currency);  
end;
```

Var

```
  Employee: TEmployee;
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## User Defined Data Types

Example

type

```
TShapeList = (Rectangle, Triangle, Circle, Ellipse, Other);  
TFigure = record  
  case TShapeList of  
    Rectangle: (Height, Width: Real);  
    Triangle: (Side1, Side2, Angle: Real);  
    Circle: (Radius: Real);  
    Ellipse, Other: ();  
end;
```

Var

```
  Figure: TFigure;
```

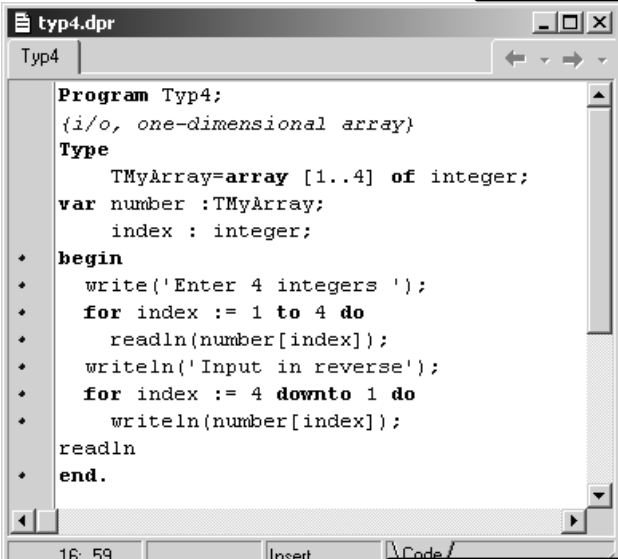
USAGE

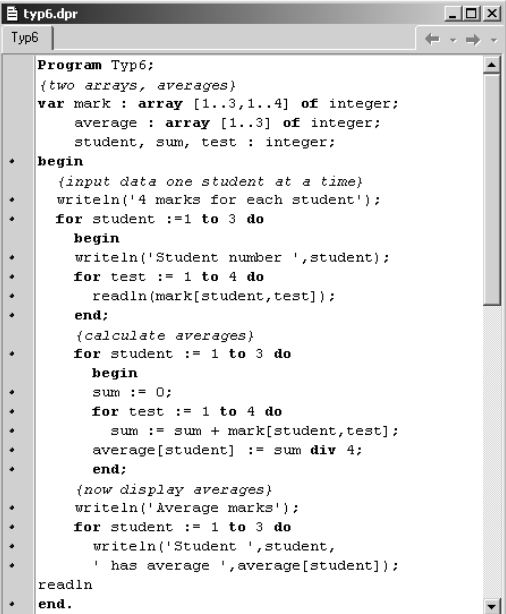
```
  Figure.Rectangle.width: =30.0;
```

Assoc.Prof.Dr.B.G.Çetiner ? 2000

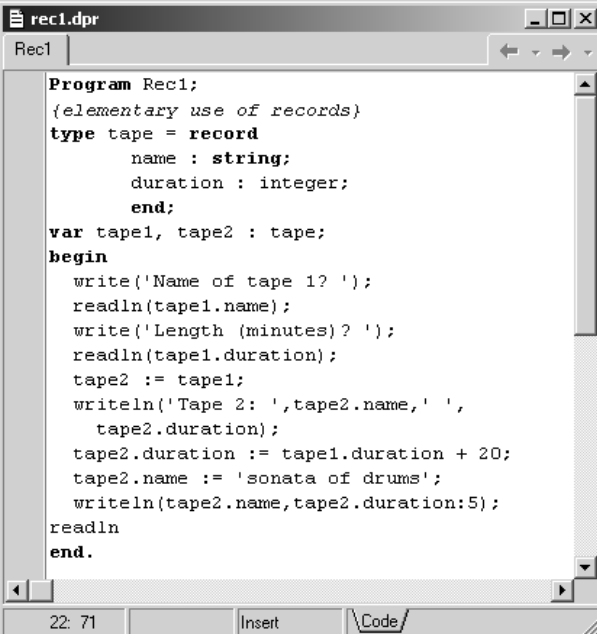
Basics of Language	User Defined Data Types
<pre> typ13.dpr typ13 program typ13; uses   sysutils,controls; type   TPerson = record     FirstName, LastName: string[40];     BirthDate: TDate;     case Citizen: Boolean of       True: (Birthplace: string[40]);       False: (Country: string[20];               EntryPort: string[20];               EntryDate, ExitDate: TDate);     end;   Var     Person:TPerson;   begin     With Person do       begin         begin           FirstName:='Gultekin';           LastName:='Cetiner';           Citizen:=false;           EntryPort:='London';           EntryDate:=StrToDate('6/6/1990');         end;         with person do           if not Citizen then             writeln(FirstName+ ' '+LastName+               ' made entry from '+EntryPort+               ' on '+DateToStr(EntryDate));         readln       end.     end. </pre>	<p data-bbox="922 510 1252 660"><b>Note: TDate is defined in controls and StrToDate/DateToStr is defined in sysutils lib.</b></p> <div data-bbox="643 667 1267 723" style="border: 1px solid black; padding: 5px; text-align: center;"> <p><i>Sample program typ13.dpr in consoleapps folder</i></p> </div> <p data-bbox="949 974 1267 996">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

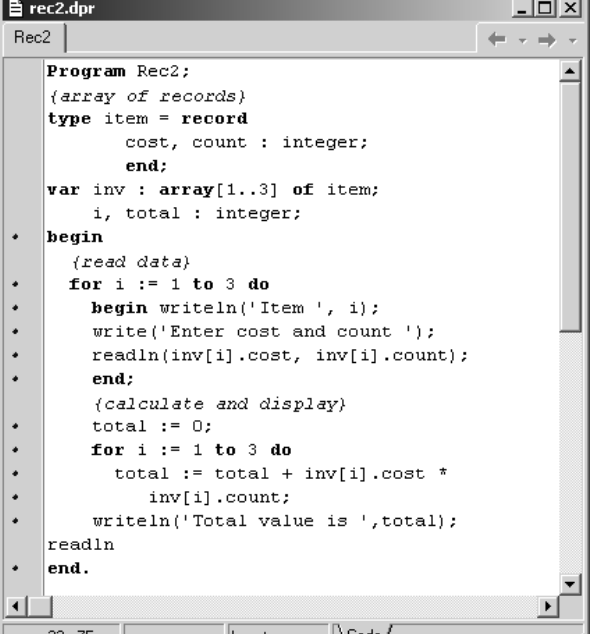
Basics of Language	User Defined Data Types
<pre> typ2.dpr Typ2 Program Typ2; {FOR, ORD, SUCC on user-defined types} type day = (mond,tues,wedn,thurs,frid,             sat,sund); var predec,succes,today : day;     ordinal : integer;   begin     for today := mond to sund do       begin         ordinal := ord(today);         if today = mond           then writeln('PRED UNDEFINED ',             'for first value - MOND')           else predec := pred(today);         if today = sund           then writeln('SUCC UNDEFINED ',             'for last value - SUND')           else succes := succ(today);         end;       readln     end. </pre>	<div data-bbox="707 1832 1267 1888" style="border: 1px solid black; padding: 5px; text-align: center;"> <p><i>Sample program typ2.dpr in consoleapps folder</i></p> </div> <p data-bbox="949 1904 1267 1926">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

Basics of Language	Array (Array Variables)
 <pre> Program Typ4; {i/o, one-dimensional array} Type   TMyArray=array [1..4] of integer; var number :TMyArray;     index : integer; begin   write('Enter 4 integers ');   for index := 1 to 4 do     readln(number[index]);   writeln(' Input in reverse');   for index := 4 downto 1 do     writeln(number[index]);   readln end. </pre>	
<p>Var number:array [1..4] of integer;</p>	<p><i>Sample program typ4.dpr in consoleapps folder</i></p> <p>Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

Basics of Language	Array (Array Variables)
 <pre> Program Typ6; {two arrays, averages} var mark : array [1..3,1..4] of integer;     average : array [1..3] of integer;     student, sum, test : integer; begin   {input data one student at a time}   writeln('4 marks for each student');   for student :=1 to 3 do     begin       writeln('Student number ',student);       for test := 1 to 4 do         readln(mark[student,test]);       end;       {calculate averages}       for student := 1 to 3 do         begin           sum := 0;           for test := 1 to 4 do             sum := sum + mark[student,test];           average[student] := sum div 4;         end;       {now display averages}       writeln('Average marks');       for student := 1 to 3 do         writeln('Student ',student,           ' has average ',average[student]);       readln     end. </pre>	
	<p><i>Sample program typ6.dpr in consoleapps folder</i></p> <p>Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>



Basics of Language	Record Types
 <pre> Program Rec1; {elementary use of records} type tape = record     name : string;     duration : integer; end; var tape1, tape2 : tape; begin     write('Name of tape 1? ');     readln(tape1.name);     write('Length (minutes)? ');     readln(tape1.duration);     tape2 := tape1;     writeln('Tape 2: ',tape2.name,' ',         tape2.duration);     tape2.duration := tape1.duration + 20;     tape2.name := 'sonata of drums';     writeln(tape2.name,tape2.duration:5);     readln end. </pre>	<p data-bbox="933 369 1252 436"><i>Sample program <b>rec1.dpr</b> in consoleapps folder</i></p> <p data-bbox="949 974 1268 996">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

Basics of Language	Record Types
 <pre> Program Rec2; {array of records} type item = record     cost, count : integer; end; var inv : array[1..3] of item;     i, total : integer; begin     {read data}     for i := 1 to 3 do         begin writeln('Item ', i);             write('Enter cost and count ');             readln(inv[i].cost, inv[i].count);         end;     {calculate and display}     total := 0;     for i := 1 to 3 do         total := total + inv[i].cost *             inv[i].count;     writeln('Total value is ',total);     readln end. </pre>	<p data-bbox="933 1299 1252 1366"><i>Sample program <b>rec2.dpr</b> in consoleapps folder</i></p> <p data-bbox="949 1904 1268 1926">Assoc.Prof.Dr.B.G.Çetiner ? 2000</p>

## Basics of Language

```
rec3.dpr
Rec3
Program Rec3;
{array of records, WITH}
type item = record
    cost, count : integer;
end;
var inv : array[1..3] of item;
    i, total : integer;
begin
    {read data}
    for i := 1 to 3 do
        begin writeln('Item ', i);
            write('Enter cost and count ');
            with inv[i] do
                readln(cost, count);
            end;
        {calculate and display}
        total := 0;
        for i := 1 to 3 do
            with inv[i] do
                total := total + cost * count;
            end;
        writeln('Total value is ', total);
    readln;
end.
```

## Record Types

*Sample program rec3.dpr  
in consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

### SUBROUTINES

1. Procedure
2. Function

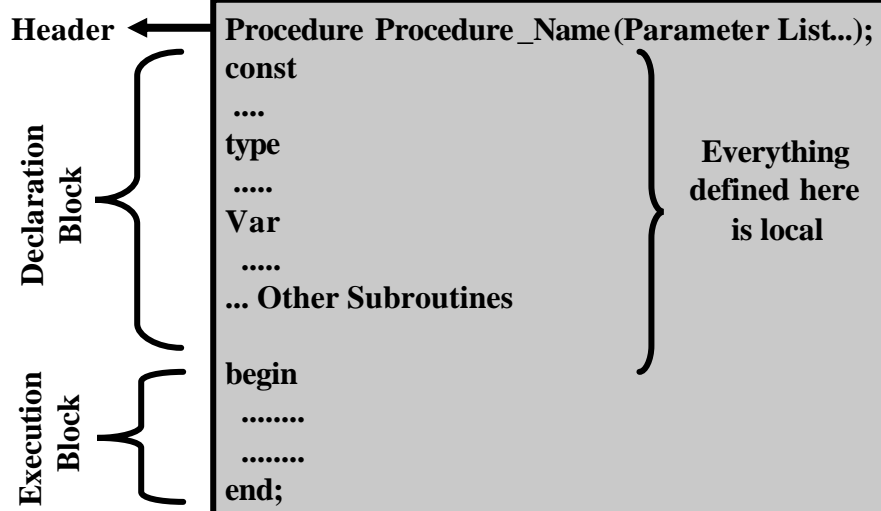
Assoc.Prof.Dr.B.G.Çetiner ? 2000

When you will define a *procedure* or *function*;

Decide

- a. Name
- b. Number of its parameters, their names and types
- c. If it is a function then returning data type.

Syntax for a procedure is given below;



Basics of Language Procedure

Examples Call By Value Call By Reference

```

Procedure GetSquare(Input:real; Var Output:real);
  Var
  Temporary:real;
begin
  Temporary:=Input*Input;
  Output:= Temporary;
end;

```

Local

Call By Value: Used to enter/pass value into the subroutine  
 Call By Reference: Used to take a value out of subroutine.  
 Write VAR in front of variable.

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Basics of Language Procedure

Examples

```

Procedure GetSquare(Input:real; Var Output:real);
  Var
  Temporary:real;
begin
  Temporary:=Input*Input;
  Output:= Temporary;
end;

```

Calling inside Program;

```

i:real;
.....
.....
GetSquare(3,i);
....

```

You have to use a variable to get the value out

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Examples

```

procedure ConvertNumberToString(N:Integer; var S:string);
var
  V: Integer;
begin
  V := Abs(N);
  S := '';
  repeat
    S := Chr(V mod 10 + Ord('0')) + S;
    V := V div 10;
  until V = 0;
  if N < 0 then S := '-' + S;
end;
    
```

To call inside Program;  
**ConvertNumberToString(123,MyString);**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Examples

Global Variable

```

Var
  Salary:real;
  ....
  .....
  .....
  No Parameter

Procedure CalculateSalary;
var
  TempSalary:real;
begin
  TempSalary:=BaseSalary*Rate;
  Salary:=TempSalary;
end;
    
```

To call from within the Program;  
**CalculateSalary;**

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc1.dpr
Proc1
Program Proc1;
{declaration and call of procedures}
var count, i : integer;
Procedure Pluses;
begin
•   writeln('++++++++++++++++++++');
•   writeln;
•   end; {pluses}
Procedure Minuses;
begin
•   writeln('-----');
•   writeln;
•   end; {minuses}
•   begin {main program}
•   write('How many times? ');
•   readln(count);
•   pluses; minuses;
•   for i := 1 to count do
•   writeln('Pascal is GREAT');
•   minuses; pluses; pluses;
readln
•   end.
```

*Sample program proc1.dpr  
in consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc2.dpr
Proc2
Program Proc2;
{procedure with parameters}
Procedure Symbols(chara : char; count : integer);
var i : integer;
begin
•   for i := 1 to count do
•   write(chara);
•   writeln;
end; {symbols}
begin {main body}
•   symbols('+',23); symbols('-',19);
•   writeln('Pascal is GREAT');
•   symbols('-',19);
•   symbols('+',23); symbols('+',23);
readln
end.
```

*Sample program proc2.dpr  
in consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc3.dpr
Proc3
Program Proc3;
{scope of variables}
var i, j : integer;
Procedure Stars(start,number:integer);
var i : integer;
begin
  writeln(number,' stars in field ',j+start-1);
  write(' ':start-1);
  for i := 1 to number do write('*');
  writeln;
end;
begin {main program}
  write('How many stars? '); readln(j);
  write('Total field width? '); readln(i);
  if i > j then
    begin
      stars(i-j+1,j);
      write(j,' stars, field width ',i);
    end
  else write('Impossible');
  readln
end.
```

*Sample program proc3.dpr  
in consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc4.dpr
Proc4
Program Proc4;
{nested procedures and scope}
var a, b : integer; {global}
Procedure x;
var product : integer;
  Procedure y;
  {multiplication}
  var j : integer; {local to y}
  begin
    product := 0;
    for j := 1 to b do product := product + a;
  end; {procedure y}
begin {body of x}
  writeln('Calculation of product');
  y;
  writeln('Product is ',product);
  write('Enter two integers '); readln(a, b);
end; {procedure x}
Procedure y;
begin
  if b<> 0 then writeln('Quotient is ',a div b)
  else writeln('No division by 0');
end; {procedure y}
begin {main program}
  write('Enter two integers '); readln(a, b);
  x; {product} y; {divide}
  readln
end.
```

*Sample program proc4.dpr in  
consoleapps folder*

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc5.dpr
Proc5
Program Proc5;
{recursive procedure}
Procedure Symb(chara : char;
               count : integer);
begin
  if count > 0
  then
    begin
      symb(chara, count-1);
      write(chara);
    end;
end; {symb}
begin {main program}
  symb('+',3); symb('-',3);
  symb('+',3); writeln;
  symb('-',3); symb('+',3);
  symb('-',3); writeln;
readln
end.
```

Sample program *proc5.dpr* in *consoleapps* folder

Assoc.Prof.Dr.B.G.Çetiner ? 2000

## Basics of Language

## Procedure

```
proc6.dpr
Proc6
Program Proc6;
{limitations of value parameters}
var x, y : integer;
Procedure Exchange(a, b : integer);
var buffer : integer;
begin
  buffer := a;
  a := b;
  b := buffer;
end; {exchange}
begin {main program}
  write('Enter two integers ');
  readln(x, y);
  exchange(x, y);
  writeln('Exchanged ', x, ' ', y);
readln
end.
```

Sample program *proc6.dpr* in *consoleapps* folder

Assoc.Prof.Dr.B.G.Çetiner ? 2000



```
Program Proc7;
{calculating power with a PROCEDURE}
var b, e, result : integer;
Procedure Power(base,exponent:integer;
                var result : integer);
var count, product : integer;
begin
  product := 1;
  for count := 1 to exponent do
    product := product * base;
  result := product;
end; {power}
begin {main program}
  write('Enter base and exponent ');
  readln(b, e);
  if e > 0 then
    begin power(b, e, result);
      writeln('Result is ',result);
    end
  else writeln('No negative exponent');
  readln
end.
```

Sample program *proc7.dpr* in *consoleapps* folder

Assoc.Prof.Dr.B.G.Çetiner ? 2000

**Function**

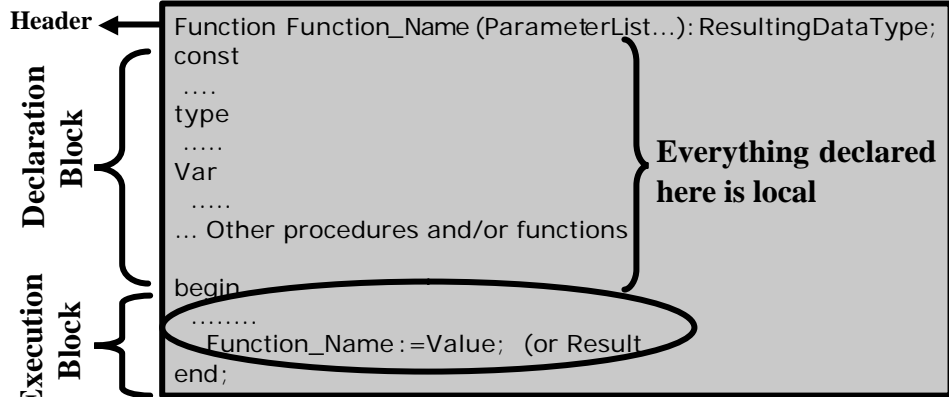
The difference between procedure and function is that the function itself returns a value whereas a procedure returns no value (like void function in C language)

e.g.  $y = \sin(x)$  → Function

e.g. `GetSin(x,SinResult);`  
`y := SinResult;` → Procedure

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Syntax for a *function* is given below;



Assoc.Prof.Dr.B.G.Çetiner ? 2000

Examples

```

Function Divide (Number1,Number2: real):Real;
begin
  if Number2 <> 0 then Divide :=Number1/Number2;
end;
    
```

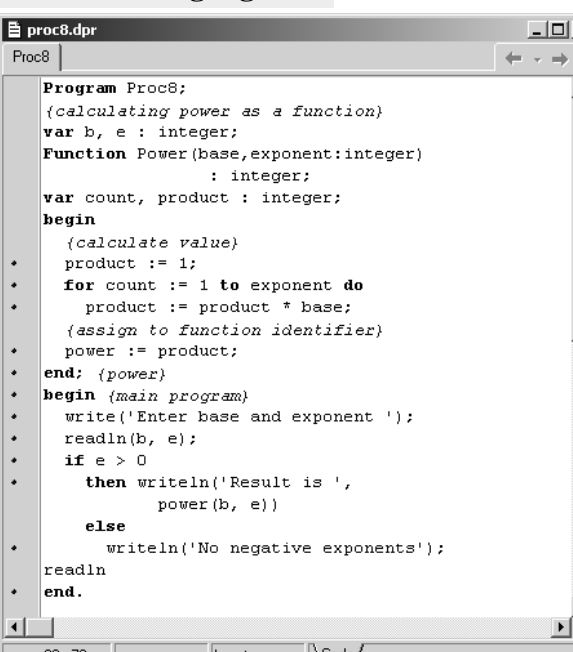
Usage

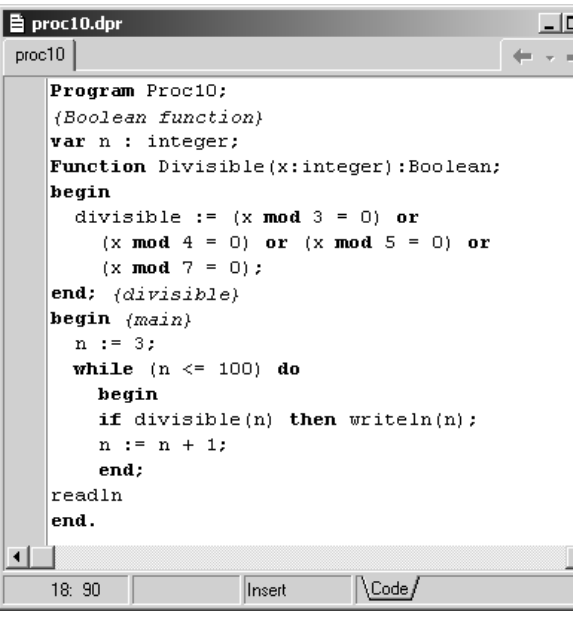
```

d:=Divide (2,3);
    
```

As seen, you can directly assign the function to a suitable value

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Basics of Language	Function
 <pre> Program Proc8; {calculating power as a function} var b, e : integer; Function Power(base,exponent:integer)            : integer; var count, product : integer; begin   {calculate value}   product := 1;   for count := 1 to exponent do     product := product * base;   {assign to function identifier}   power := product; end; {power} begin {main program}   write('Enter base and exponent ');   readln(b, e);   if e &gt; 0   then writeln('Result is ',               power(b, e))   else     writeln('No negative exponents');   readln end. </pre>	<p style="margin: 0;"><i>Sample program <b>proc8.dpr</b> in consoleapps folder</i></p>
Assoc.Prof.Dr.B.G.Çetiner ? 2000	

Basics of Language	Function
 <pre> Program Proc10; {Boolean function} var n : integer; Function Divisible(x:integer):Boolean; begin   divisible := (x mod 3 = 0) or               (x mod 4 = 0) or (x mod 5 = 0) or               (x mod 7 = 0); end; {divisible} begin {main}   n := 3;   while (n &lt;= 100) do     begin       if divisible(n) then writeln(n);       n := n + 1;     end;   readln end. </pre>	<p style="margin: 0;"><i>Sample program <b>proc10.dpr</b> in consoleapps folder</i></p>
Assoc.Prof.Dr.B.G.Çetiner ? 2000	

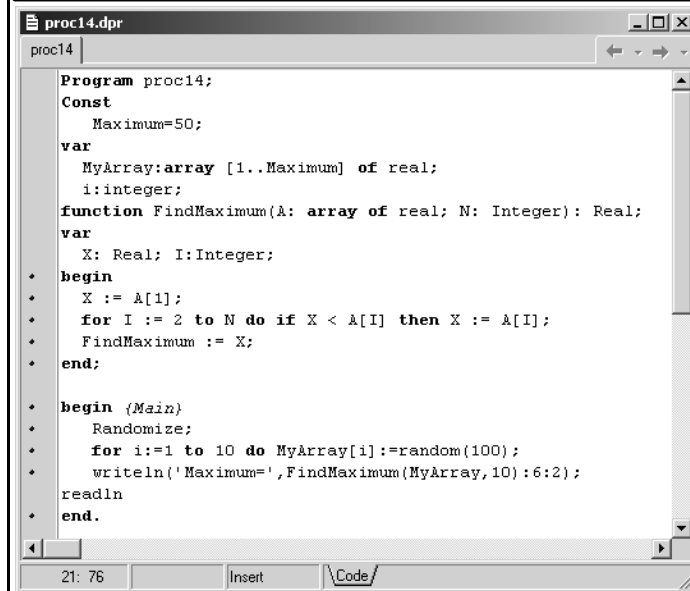
Basics of Language	Function
<div style="border: 1px solid gray; padding: 5px;"> <div style="display: flex; justify-content: space-between; align-items: center;"> <span>proc11.dpr</span> <span style="border: 1px solid black; padding: 2px;">Sample program <i>proc11.dpr</i> in consoleapps folder</span> </div> <div style="border: 1px solid gray; padding: 5px; min-height: 200px;"> <pre> proc11 Program Proc11; {Boolean function in conditions} var n : integer; Function Divisible(x:integer):Boolean; begin     divisible := (x mod 3 = 0) or                 (x mod 4 = 0) or (x mod 5 = 0) or                 (x mod 7 = 0); end; {divisible} begin {main}     n := 3;     while divisible(n) do         begin write(n:4); n := n + 1;               end;     readln end.</pre> </div> <div style="display: flex; justify-content: space-between; align-items: center; font-size: small;"> <span>16: 95</span> <span>Insert</span> <span>Code</span> </div> </div>	

Assoc.Prof.Dr.B.G.Çetiner ? 2000

Basics of Language	Function
<div style="border: 1px solid gray; padding: 5px;"> <div style="display: flex; justify-content: space-between; align-items: center;"> <span>proc12.dpr</span> <span style="border: 1px solid black; padding: 2px;">Sample program <i>proc12.dpr</i> in consoleapps folder</span> </div> <div style="border: 1px solid gray; padding: 5px; min-height: 200px;"> <pre> Proc12 Program Proc12; {example of char function} var c : char; Function To_caps(chara:char):char; begin     if (chara &gt;= 'a') and (chara &lt;= 'z')     then {convert to code, shift, convert}         to_caps := chr(ord(chara) +                     ord('A') - ord('a'))     else to_caps := chara; end; {to_caps} begin {main}     repeat         write('Type any character,',               ' \$ to stop ');         readln(c);         writeln(to_caps(c));     until c = '\$';     readln end.</pre> </div> <div style="display: flex; justify-content: space-between; align-items: center; font-size: small;"> <span>20: 76</span> <span>Insert</span> <span>Code</span> </div> </div>	

Assoc.Prof.Dr.B.G.Çetiner ? 2000

*Sample program proc14.dpr in consoleapps folder*



```
Program proc14;
Const
  Maximum=50;
var
  MyArray:array [1..Maximum] of real;
  i:integer;
function FindMaximum(A: array of real; N: Integer): Real;
var
  X: Real; I:Integer;
begin
  X := A[1];
  for I := 2 to N do if X < A[I] then X := A[I];
  FindMaximum := X;
end;
begin (Main)
  Randomize;
  for i:=1 to 10 do MyArray[i]:=random(100);
  writeln(' Maximum=', FindMaximum(MyArray,10):6:2);
readln
end.
```

Assoc.Prof.Dr.B.G.Çetiner ' 2000