

"

,

|||

, 2008.



..... 4

..... 5

..... 5

..... 7

..... 9

Windows- 11

..... 16

..... 22

..... 23

..... 25

..... 28

..... 29

..... 29

..... 32

..... 34

String 35

..... 37

..... 38

IF 38

Case 39

..... 41

FOR - 41

REPEAT - 42

WHILE - 43

..... 45

..... 47

..... 47

..... 48

..... 49

..... 51

..... 52

, 53

..... 55

..... 55

..... 57

-
-

-
-
-
-

- *algol* - ALGORitmic Language,
- *fortran* - FORMula TRANslation system,
- *cobol* - COMmon Business Rriented Language,
- *pl/1* - Program Language no. 1,
- *c* -
- *ada* -
- *gps* - General Problem Solver,
- *simula* -
- *pascal*

- *basic* - Beginners All-purpose Symbolic Instruction Code,

- *lisp* -
- *prolog* -

- *planer*

()

- *RPG, Mars, Arius, Proza, Graph.*

60.

1965.

WG2.1

60

, 1969/70. 68 (1968.

).

(*Blaise Pascal*, 1623-1662.)

Wirth),

(*Niclaus*

()

()

(, , ,).

1979.

”

“

Windows

Borland “ ” 1994.
Delphi

Object Pascal.

(Application Programming Interface -)

Visual Basic (),

Windows NT,

Windows 3.1x

Windows 95
16-
, 32-

16-

32-

2.

255
(multithreading),
SQL

95

1996.

1997.

1 2).

Active X

(. *bj*)

4 1998.

Active X

5
6

7
Windows XP

6
Rave Designer.

Delphi 2009

64-

(native code),

(. *x*)

(, , *.dll*),
, BDE (Borland Database Engine), , Report Smith .
 :
 275 .
 , (80 , 90-
 80 - 500). 3.0
(.dpl - Delphi Package Library)
(.dll).
 . , . ,
 .
 : , , .
 .
 , , , ,
 ++.
 , , , ,
 .

Windows-

(*form*).

, *main window*.
 , *secondary windows*.

, *dialog box*.

(*modal*)

(*modeless*)

, *child window*,
 , *parent window*,

, *mouse cursor*.

- - *click*
- - *drag and drop*.

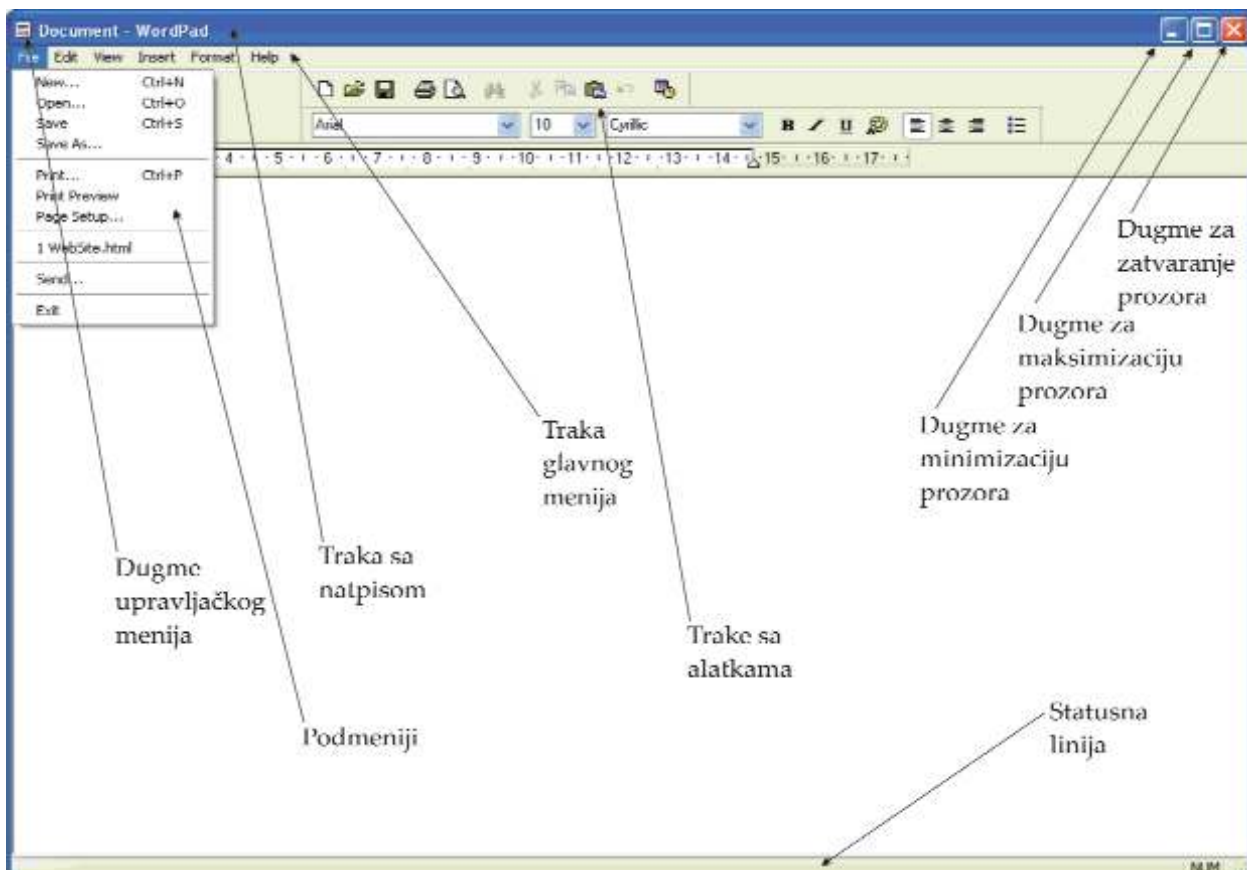
, *text cursor.*

Enter Esc.

Alt Ctrl,

- *accelerator character.*

WordPad,

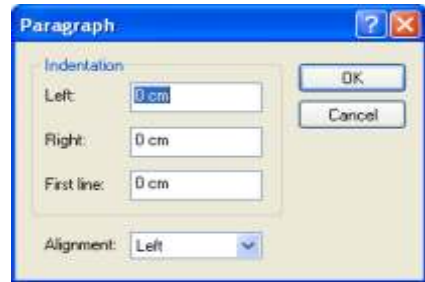


1. _____ - *Caption bar.*

-
-
-

↑ ↓ → ←

(...)



-
-

- *Cancel*

Yes No.

Help.

- *shortcut keys.*

Ctrl, *1* *12* *Ctrl.* *Shift, Alt*

- *control menu*

- *system menu.*

Maximize,

- *Size*

- *Close.*

- *Restore,*

Minimize,

- *Move,*

• - *label.*

- *Caption*

• - *button*.

• - *edit box*.

• - *check box*.

(✓)

• - *radio button*.

(⊙)

(○).

- *dimmed*.

(*focus*).

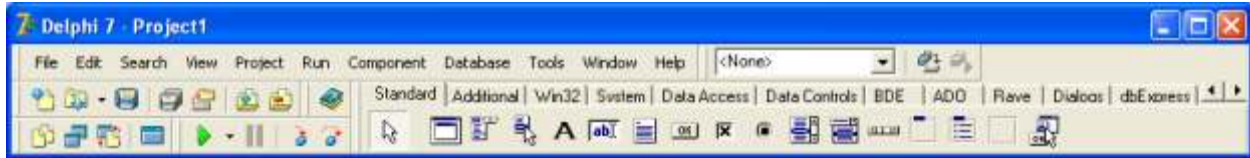
- :
- ;
-

Shift + Tab

Tab (↵)

• (Single Document Interface Applications -
). (,),

• (Multiple Document Interface Applications -
).



(Delphi 7 - Project1)

(Control Menu Button)

(): Minimize, Restore Down / Maximize Close.

Project, Run, Component, Database, Tools, Window Help. : File, Edit, Search, View,

File

*New
Open*

Application

*Open Project
Reopen*

Save

Save As

Save Project As

*Save All
Close*

*Close All
Use Unit
Print
Exit*

Edit

:

Undo/Undelete
Redo
Cut
Copy
Paste
Delete
Select All

clipboard-
clipboard

clipboard

Search

:

Find
Replace
Search Again

, ,
, ,

View

:

Object Inspector
Object TreeView
Alignment Palette
Toggle Form/Unit

Object Inspector
Object TreeView
Alignment Palette
form unit

Run

:

Run
Program Reset

()

) (

- speed buttons

(,),

• Desktop
debug desktop

Desktop speedsetting

: Save current desktop Set

• Standard
from project

: New, Open, Save, Save All, Open project, Add file to project, Remove file

• Custom

: Help contents

• View
• Debug

: View unit, View form, Toggle Unit/Form, New form
: Run, Pause, Trace into, Step over

(,).

- **Component palette** : *Standard, Additional, Win32, System, Data Access, Data Controls, dbExpress, DataSnap, BDE, ADO, InterBase, WebServices, InternetExpress, Internet, WebSnap, Decision Cube, Dialogs, Win 3.1, Samples, ActiveX, Rave, Indy Clients, Indy Servers, Indy Intercepts, Indy I/O Handlers, Indy Misc, COM+, InterBase Admin, IW Standard, IW Data, IW Client Side, IW Control, Servers*

Standard

Additional, Win32, System.

Standard

- *Label* ()
- *Edit* ()
- *Button* ().

(Label)

(Edit)

(Button)

Standard

- *m* (),
- *ComboBox* ()
- *Panel* ().

Additional

- *BitBtn* (),
- *Image* ()
- *LabeledEdit* ().

Win32

DateTimePicker

System

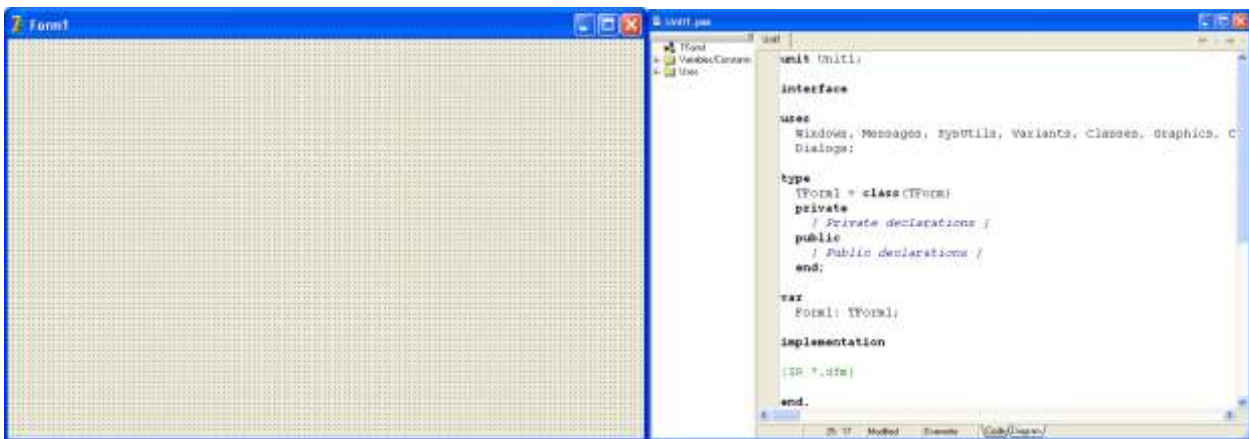
Timer

Form1.

- *(Properties)*
- *Form1,*

Font

: *Minimize, Restore Down / Maximize Close*



Unit1.

Code

Explorer

Edit Window,

F12.
Ctrl+F12,
Shift+F12.

- *Object Tree View*
- *Object Inspector.*

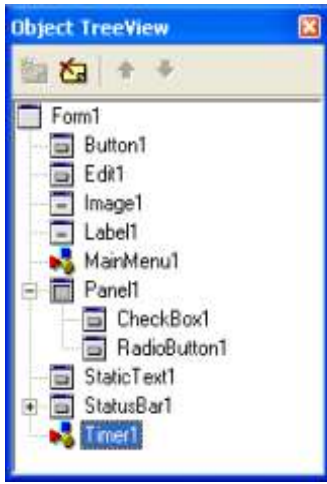
X.

Inspector) Shift+Alt+F11 (Object TreeView).

F11 (Object

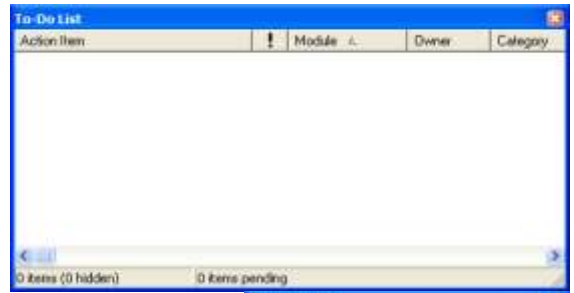
Object TreeView

Object TreeView



To - Do List (

(
);



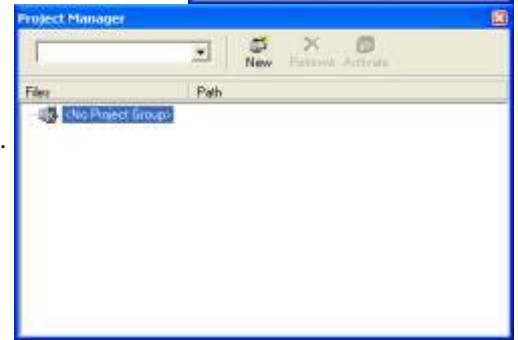
Components (

);



Project Manager

j



Align (

);

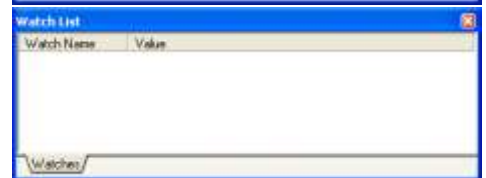
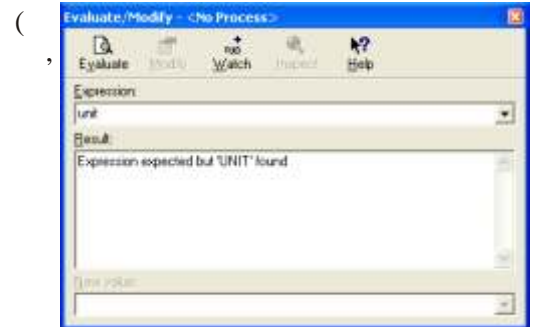


, *Debug*

Evaluate/Modify

Watch

),



Delphi



•
:
: 0,1,2,3,4,5,6,7,8,9
+ - * / = ^ < > () [] {} . , ; ' # \$
:
•
*
/
+
-
()
•
:=
:
;
ASCII
\$
(* *)
{ }
+
-
,
•
=
<
>
<=
>=
<>
•
^
(.)
[]
()
..
()

and	array	as	asm	begin	constructor
case	class	const	div	destructor	dispinterface
do	downto	else	end	except	exports
file	finalization	finally	for	function	goto
if	implementation	inline	in	inherited	initialization
interface	is	label	library	mod	nil
not	object	of	or	out	packed
procedure	program	property	raise	record	resourcestring
repeat	set	shl	shr	string	then
thread	var to	try	type	unit	until
uses	var	while	with	xor	

integer ,
integer .
 () :

- ;
- ,
- .
- **eol** (end-of-line,)
- //
- { ... }
- **space** ()

255 , 255- (255
 255).

-
-
-
- **GOTO** ()
-
-
- (*If* , *Case*)
 (*For* , *Repeat* , *While*)

promeljiva := vrednost;
 :=
 (*Read* , *Write*)

GOTO
 ()

End. **Begin,**

begin <naredbe>
 end;
 ()


```

:
type dan = (ponedeljak, utorak, sreda, cetvrtak, petak, subota, nedelja);
var danas: dan;
const e=2.718281827;
label kraj, ovde, 10;
function TreciKoren(x: real): real;
procedure Zameni(a, b: integer);

```

begin.

```

begin { ( .begin), ( .end):
...
end.

```

- *Project1.cfg* -
- *Project1.dpr* - Delphi Project
- *Project1.res* -
- *Project1.dof* -
- *Unit1.pas* -
- *Unit1.dfm* - Delphi Form

(*Unit1.dcu, Unit1.ddp*) .

Project1.exe -

Project1.dpr

```

program Project1;
uses
  Forms,
  Unit1 in 'Unit1.pas' {Form1};
{$R *.res}
begin
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.Run;
end.

```

Unit1.dfm

```

object Form1: TForm1
  Left = 192
  Top = 114
  Width = 870
  Height = 640
  Caption = 'Form1'
  Color = clBtnFace
  Font.Charset = DEFAULT_CHARSET
  Font.Color = clWindowText

```

```
    Font.Height = -11
    Font.Name = 'MS Sans Serif'
    Font.Style = []
    OldCreateOrder = False
    PixelsPerInch = 96
    TextHeight = 13
end
```

Unit1.pas :

```
unit Unit1;
```

```
interface
```

```
uses
```

```
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs;
```

```
type
```

```
    TForm1 = class(TForm)
```

```
    private
```

```
        { Private declarations }
```

```
    public
```

```
        { Public declarations }
```

```
    end;
```

```
var
```

```
    Form1: TForm1;
```

```
implementation
```

```
    {$R *.dfm}
```

```
end.
```


(char) *ASCII*

ASCII

256

/

- *Pred* -
- *Succ* -
- *UpCase* -

(

- *Ord* -
- *Chr* -

ASCII

0 255

ASCII

:

-
-

ASCII

(integer)

Integer

- *Pred*
- *Succ*
- +
- -
- *
- *Div*
- *Mod*
- *Abs*
- *Sqr*

:

- *Ord* -
- *Chr* -

ASCII

0 255

ASCII

(/)

+, - *

127

-128

1,

- **Integer** -2147483648 .. 2147483647, .
- **Cardinal** 0 .. 4294967295, .

ShortInt	-128 .. 127	8
SmallInt	-32768 .. 32767	16
LongInt	-2147483648 .. 2147483647	32
Int64	$-2^{63} .. 2^{63}-1$	64
Byte	0 .. 255	8
Word	0 .. 65535	16
LongWord	0 .. 4294967295	32

- **StrToInt(*tekst*)** -

- **StrToIntDef(*tekst*,*broj*)** -

- **IntToStr(*broj*)** -

- **Str(*broj*, *tekst*)** -

- **Val(*tekst*, *broj*, *kod_greske*)** -

```

var ImePromenljive:TipPromenljive;
var a,b,c,d:integer;
var a:integer; b:integer; c:integer; d:integer;
var a:integer;
    b:integer;

```


(Real)

mantisaE+eksponent

()

real

$5.0 \times 10^{-324} .. 1.7 \times 10^{308}$

15-16

Real48	$2.9 \times 10^{-39} .. 1.7 \times 10^{38}$	11 - 12	6
Single	$1.5 \times 10^{-45} .. 3.4 \times 10^{38}$	7 - 8	4
Double	$5.0 \times 10^{-324} .. 1.7 \times 10^{308}$	15 - 16	8
Extended	$3.6 \times 10^{-4951} .. 1.1 \times 10^{4932}$	19 - 20	10
Comp	$-2^{63} .. 2^{63-1} (-9223372036854775808..9223372036854775807)$	19 - 20	8
Currency	-922337203685477.5808 .. 922337203685477.5807	19 - 20	8

- **Real48**
- **Single Double**
- **Extended**
- **Comp**
- **Currency**

64 -
()

10.000).

comp

15

,14

Math

```

Unit.pas). j ( uses
:
• +
• -
• *
• /
• frac , , 0
• int , ,
• abs
• sin
• cos
• arctan
• ln
• exp
• sqr
• sqrt
• random 0 1( ).

( ) ,
Randomize Random.

:
• round
• trunc , , , ,
, , , ,
Frac, Int, Round, Trunc, Abs, Sin, Cos, ArcTan, Ln, Exp, Sqr Sqrt
( ).
( ) :
• FloatTo Str -
• StrToFloat -
• StrToFloatDef - StrToInt
StrToIntDef
) :
• Str(broj, tekst) - tekst. broj
• Val(tekst, broj, kod_greske) - tekst broj.
kod_greske
:
VAR a,b,c:real;

```

• *pi* -

3.1415926535897932385

:

•

100,

•

•

15, 25, 35 45

5,

•

)

a b.

(

•

(

h.

•

(

a, b c.

r.

•

t

•

r.

•

•

b

D.

•

s.

, *true false.*

/ :

• *not*

• *or*

• *and*

• *xor*

• *pred*

• *succ*

• *odd*

• *ord*

(*Ord(false) = 0, Ord(true) = 1*).

false.

• <

• <=

• <>

• =

• >=

• >

VAR b:STRING[11];

- +
- *copy(a,n,m)*
- *length(a)*
- *pos(b,a)*
- *concat(a,b,..)*
- *delete(a,n,m)*
- *insert(b,a,n)*
- *val(tekst, broj, kod_greske)*
- *str(broj, tekst)*
- *LowerCase* -
- *UpperCase* -

- (set).

255 (*boolean, char, integer*).

```

TYPE slova = SET OF 'a'..'z';
cifre = SET OF 0..9;
mese = (jan,feb,mart,apr,maj,jun,jul,aug,sept,okt,nov,dec);
godina = SET OF mese;

```

```

VAR a:slova;
cifra:cifre;
znak:SET OF (pik,karo,herc,tref);
ceobroj:SET OF 0..255;

```

```

• ( +),
• ( *)
• ( -)

• ( =),
• ( <>),
• ( < <=)
• ( > >=)
• ( in)

```

:=[];

:= +[];

:=[' ',' ',' ',' ',' ',' ',' ',' '];

```

_____ ( a=b), _____
( a<b a<=b) , ( a<>b).
( a<b) ( a>b a>=b).
( a>b)

```



GOTO
:

- *If*
- *Case*

256

IF

```
IF <uslov> THEN <naredba>
    ELSE <naredba>;
```

IF

Then, *Then* *Else.*

```
IF <uslov>
    THEN Begin <naredbe>
           End
    ELSE Begin <naredbe>
           End;
```

Else *Then* *Else.*

Then , *Else* ,

Else ;

Then, *IF* , *Then*

IF *Else.*

IF *IF* *Else,*

IF :

```
IF <uslov1>
    THEN Begin IF <uslov2> THEN <naredba1>
           End
    ELSE <naredba2>;
<naredba3>;
```



```

      :
      <slucajN> : <naredbaN>;
    ELSE Begin <naredbe>
      End
  END;

```

```

      CASE
      ,
      CASE
      END
      ELSE
      ELSE
      ELSE
      CASE
      CASE
      IF
      ;
      :
      IF
      CASE <izraz> OF
      <slucaj1> : CASE <izraz> OF
        <slucaj1> : <naredbe>;
        <slucaj2> : <naredbe>;
        :
        <slucajN> : <naredbe>;
        ELSE <naredbe>
      END;
      <slucaj2> : <naredbe>;
      :
      <slucajN> : If <uslov>
        then <naredba>
        else <naredba>
      :
      <slucajN> : If <uslov> then <naredba>;
      ELSE <naredbe>
  END;

```

•
•
•
•
•
•
•
•
•
•
•
•
•
•
•

20. 21.

• m n n .

• ().

• ().

• n (

• n (

• n (

• n (

• n (

REPEAT-

For , , **Repeat ...**

```
until...
:
REPEAT <naredba1>;
    <naredba2>;
    ...
    <naredbaN>;
UNTIL <uslov>;
```

, , (

, ,).

For Repeat For Repeat.

```
:
REPEAT <naredba>;
    ...
    <naredba>;
UNTIL (uslov1)or(uslov2)and(uslov3);

( 1 2 3 ).
REPEAT <naredba>;
    ...
    <naredba>;
UNTIL ((uslov1)or(uslov2))and(uslov3);

( 3 2 1 ).
```



```
function Stepen(b,e:real):real;
```

```
function TForm1.Stepen(b,e:real):real;
var s:real;
begin s:=1;
      While e>0 do
      begin s:=s*b; e:=e-1;
            end;
      stepen:=s;
end;
```

b e

```
a:=Stepen(b,n); <-
a:=Stepen(b,3); <-
a:=Stepen(2,4); <-
```

```
procedure Stepen(b,e:real;VAR s:real);
```

VAR.

```

procedure TForm1.Stepen(b,e:real; VAR s:real);
begin s:=1;
      While e>0 do
        begin s:=s*b; e:=e-1;
              end;
end;

```

b, e s

```

Stepen(b,n,s);      <-
a:=Stepen(b,3,s);  <-
a:=Stepen(2,4,s);  <-

```

2

/

-
-
-
-
-
-
-
-
-
-
-

n-

n.
1000

n.

n.

n.

n

0.



•
•
•
•

(boolean, char, integer, ...).

ARRAY [1..100] of REAL

ARRAY ['a'..'z'] of INTEGER
26

array

(
TYPE NIZ=ARRAY[1..50] of INTEGER;
ZNAK=(pik,karo,herc,tref);
BROJ=1..14;
SPIL=ARRAY[znak] of broj;
VAR a:INTEGER;
b:NIZ;
karta:SPIL;

"Error 22: Structure too large"
"Error 96: Too many variables"


```

TYPE adrese=RECORD ulica,broj,mesto,drzava:string;
      END;
      brojtlf=RECORD pozivni:string;
      broj:string;
      END;
      adresar=RECORD prezime,ime:string;
      adresa:adrese;
      telefon:brojtlf;
      END;

```

():

```

VAR ucenik1,ucenik2:RECORD ime,prezime,skola:string;
      razred:(I,II,III,IV);
      odeljenje:1..8;
      END;
      podatak1,podatak2:adresar;
      telefon:brojtlf;

```

```

podatak1.prezime
podatak1.adresa.ulica

```

```

ucenik1.prezime<ucenik2.prezime
podatak1.adresa.ulica<=podatak2.adresa.ulica

```

```

ucenik1<ucenik2
podatak1.adresa<=podatak2.adresa

```

```

VAR a,b,pom:slog;

```

a b

pom

```

pom:=a; a:=b; b:=pom;

```

a b,

a ba:

podatak1.pozicija:=a

podatak1

(*podatak1*).

With

```
podatak1.prezime:='Ilic';  
podatak1.ime:='Miroslav';  
podatak1.adresa:='Ulica i broj';  
podatak1.telefon:='011/123-45-67';
```

```
with podatak1 do  
begin prezime:='Ilic';  
       ime:='Miroslav';  
       adresa:='Ulica i broj';  
       telefon:='011/123-45-67';  
end;
```

- *Append*
- *AssignFile*

- *CloseFile*
- *Eof*

- *Eoln*

- *Erase*
- *FilePos*

- *FileSize*

- *IOResult*

- *Read*
- *Readln*

- *Rename*

- *Reset*
- *Rewrite*

- *Seek*

- *SeekEof*

- *SeekEoln*

- *Truncate*

- *Write*
- *Writeln*

```

TYPE imena=FILE OF string;
     adresa=RECORD ulica:string;
                   broj:integer
           END;
     adrese=FILE OF adresa;

```

TextFile.

```

VAR imenik:imena;
     adresar:adresa;
     ceobroj:FILE OF integer;
     slova:FILE OF char;
     opis:TextFile;

```

```
AssignFile(imenik, 'imena.dat');
```

imenik

imena.dat

```
AssignFile(opis, 'pismo.txt');
```

opis

pismo.txt

ReWrite:

```
ReWrite(imenik);
```

imena.dat

```
ReWrite(opis);
```

pismo.txt

Reset:

```
Reset(imenik);
```

imena.dat

```
Reset(opis);
```

pismo.txt

IOResult

Reset

0

```

{$I-} Reset(imenik); {$I+}
If IOResult<>0 then ReWrite(imenik);

```

```

    {$I-}                                I/O (Input/Output,
                                         I/O

```

```

    Reset(imenik) " "

```

```

    {$I+}                                I/O

```

```

IOResult                                IOResult
ReWrite(imenik), .                    0, I/O then,

```

```

{$I-} Append(opis); {$I+}
If IOResult<>0 then ReWrite(opis);

```

IOResult.

```

{$I-} Reset(imenik); {$I+}
i:=IOResult;
If IOResult<>0 then ReWrite(imenik);

```

```

I/O                                IOResult imenik if
Reset.

```

Write WriteLn:

```

Write(imenik,ime);
    imena.dat

```

```

WriteLn(opis,'tekst izme u apostrofa ili promenljiva tipa string');
Append(opis);

```

pismo.txt

```

Read(ime);

```

```

ime:=a;
ime:='Viktorija';
ime:=a+b;

```

```

    CloseFile:
CloseFile(imenik);

    Append.
Read(imenik, ime)
    imena.dat
Read(opis, ime)
    pismo.txt

    EOF:
While not EOF(imenik) do Read(imenik, ime);
While not EOF(opis) do ReadLn(opis, ime);

    for FileSize (
) :
For i:=1 to FileSize(imenik) do Read(imenik, ime);

    Write WriteLn (
, );

Write(ime);
WriteLn(ime);

    Reset
(
,
);
:
:
• ( )
•

    Seek:
Seek(imenik, redbrpod);
Read(imenik, ime)
    redbrpod
:

While not EOF(imenik) or (Pos(sifra, ime) <> 0) do Read(imenik, ime);
sifra
,
,
,

```

```

Reset(imenik);

For i:=1 to FileSize(imenik)-1 do
  For j:=i+1 to FileSize(imenik) do
    Begin Seek(imenik,i); Read(imenik,ime1);
           i
           Seek(imenik,j); Read(imenik,ime2);
           j
    If ime1>ime2 then
      Begin Seek(imenik,j); Write(imenik,ime1);
           j
           Seek(imenik,i); Write(imenik,ime2);
           i
      End
    End;
End;

```

```

Erase(imenik)

```

Erase:

```

Reset(imenik);
ReWrite(pom);
For i:=1 to FileSize(imenik) do
  Begin Read(imenik,ime);
       If ime<>'sifra' then Write(pom,ime)
  End;
Close(pom);
Erase(imenik);
Rename(pom,'imena.dat');

```

“ ”

sifra

```
Reset(imenik);
For i:=1 to FileSize(imenik) do
Begin Read(imenik,ime);
  If ime<>sifra then
  Begin Seek(imenik,i-1);
    Write(imenik,ime)
  End
End;
Truncate(FileSize(imenik)-1);
```

Truncate

Truncate

Seek).

Rename