Primele programe, instrucţiuni şi proceduri

Unul din cele mai simple programe, este acela de a afişa un text pe ecran, ca în exemplul următor **:**

```
Program inceput;
begin
writeln('Vornicescu Silviu');
writeln('Programator al firmei Delphin Software');
end.
```

În acest caz programul cuprinde **:**

- Un **antet**, adică cuvîntul rezervat **Program**, urmat de numele programului. În **Pascal** folosirea antetului este obligatorie, dar în **Turbo Pascal** antetul nu mai este obligatoriu, dar e bine să îl folosim pentru o mai mare claritate a programelor.
- **Blocul** programului, singurul care este obligatoriu. În interiorul blocului se pot face declaraţii de variabile sau constante şi pot fi scrise instrucţiuni. Un bloc poate să conţină la rîndul său alte blocuri.

Următorul program este asemănător cu primul, dar conţine elemente noi în structura sa.

```
uses crt;
begin
clrscr;
writeln('Salut');
end.
```

În acest caz, programul conţine în plus cuvîntul rezervat **Uses**, urmat de numele unit-ului care se va utiliza, **crt;**

În acest exemplu este utilizată subrutina **clrscr (cl**ea**r scr**een**)** din unitul **crt ,** care este folosită pentru ştergerea ferestrei curente cu culoarea fondului. Implicit, culoarea fondului este neagră, iar a scrisului este albă.

Scrierea datelor

Orice operaţie de scriere a datelor se execută începînd din poziţia curentă a cursorului (liniuţa care clipeşte) şi lasă cursorul după ultimul caracter scris.

Procedurile Write şi WriteLn

Sintaxa : Write (argument1, argument2, …, argument n)

Numărul minim de argumente este unu. Un argument poate fi afişat cu sau fără format. O afişare cu format are forma **: e:w:f** în care

- **e** reprezintă expresia a cărei valoare se afişează
- pe **w** poziţii sau pe mai multe, dacă valoarea de afişat depăşeşte **w** poziţii
- valoarea expresiei **f** se referă la numărul zecimalelor cu care se va scrie valoarea expresiei **e** de tip real, în reprezentare fără exponent.

Valoarea argumentelor se scrie pe linia curentă, fără spaţii între ele. În funcţie de tipul expresiei, formele admise ale argumentului pot fi cu sau fără format special.

Argumentul **"e"** poate fi de tip **CHAR (caracter)**, **STRING (şir de caractere)**, **BOOLEAN (logic)** sau **INTEGER (Număr întreg).** Formele posibile sînt **: e** sau **e:w**

Argumentul **"e"** poate fi şi de tip **REAL.** Formele posibile sînt **: e** sau **e:w** sau **e:w:f**.

Exemple de afişare date cu Write	
Write('1234567890');	Afişează 1234567890
Write(1969:10);	Afişează numărul 1969 începind cu coloana 10 spre coloana 7, pentru că numarul este aliniat la dreapta.
Write(9.752:7:2);	Afişează 9.75 începind cu coloana 7 spre coloana 4. O coloană este ocupată de punctul zecimal, iar afişarea se face cu 2 zecimale. Numărul este aliniat la dreapta.
Write(9.999:7:2);	Afişează 10.00 începind cu coloana 7 spre coloana 3. O coloană este ocupată de punctul zecimal, iar afişarea se face cu 2 zecimale. Se observă că numărul este rotunjit superior. Numărul este aliniat la dreapta.
Write(3.14:7);	Afişează 3.1E+00 . Dacă nu se specifică numărul de zecimale pentru afişarea unui numar real, se afişează în format ştiinţific.

Procedura **WriteLn** are apelul cu aceiaşi formă cu a procedurii **Write**. Dacă este apelată fără parametri, are ca efect trecerea cursorului la începutul liniei următoare. Dacă există o listă de parametri, efectul este identic cu al procedurii **Write**, urmată de apelul procedurii **WriteLn** fără argumente.

Exemple de afişare date cu WriteLn	
WriteLn;	Afişează un rînd gol şi mută cursorul la începutul liniei următoare.

WriteLn('Vornicescu Silviu');	Afişează textul dintre apostroafe şi mută cursorul la începutul liniei următoare. Textul este aliniat la stînga.

Listingul programului Prg_0001

```
program inceput_Prg_0001;
var a,b : integer;              {Se declară variabilele "a" şi "b" ca nr. întregi}
begin                           {Începutul programului}
WriteLn('Vornicescu Silviu');   {Afişează textul dintre apostroafe şi mută
                                cursorul la începutul liniei următoare}
WriteLn('Programator al firmei Delphin Software'); {Afişeaza textul dintre
                                apostroafe şi mută cursorul la inceputul liniei
                                următoare}
WriteLn;                        {Afişează un rînd gol şi mută cursorul la începutul
                                liniei următoare}
Write('1234567890');            {Afişează 1234567890}
WriteLn;Write(1969:10);         {Trece cursorul pe rîndul următor şi apoi
                                afişează numărul 1969 începînd cu coloana 10
                                spre coloana 7, pentru că numărul este aliniat
                                la dreapta.}
WriteLn;Write(1969:9);          {Trece cursorul pe rîndul următor şi apoi
                                afişează numărul 1969 începînd cu coloana 9
                                spre coloana 6, pentru că numărul este aliniat
                                la dreapta.}
WriteLn;Write(1969:8);          {Trece cursorul pe rîndul următor şi apoi
                                afişează numărul 1969 începînd cu coloana 8
                                spre coloana 5, pentru că numărul este aliniat
                                la dreapta.}
WriteLn;Write(9.752:7:2);       {Afişează 9.75 începînd cu coloana 7 spre coloana
                                4. O coloană este ocupată de punctul zecimal, iar
                                afişarea se face cu 2 zecimale. Numărul este
                                aliniat la dreapta.}
WriteLn;Write(9.999:7:2);       {Afişează 10.00 începînd cu coloana 7 spre coloana
                                3. O coloană este ocupată de punctul zecimal, iar
                                afişarea se face cu 2 zecimale. Se observă că
                                numărul este rotunjit superior. Numărul este
                                aliniat la dreapta.}
WriteLn;Write(7.2569:7:3);      {Afişează 7.257 începînd cu coloana 7 spre coloana
                                3. O coloană este ocupată de punctul zecimal, iar
                                afişarea se face cu 3 zecimale. Se observă ca
                                numărul este rotunjit superior. Numărul este
                                aliniat la dreapta.}
WriteLn;Write(3.14:7);          {Afişează 3.1E+00 . Dacă nu se specifică numărul
                                de zecimale pentru afişarea unui numar real, se
                                afişează în format ştiinţific.}
a:=5;                           {Se atribuie valoarea 5, variabilei "a"}
```

```
WriteLn;WriteLn(a);          {Se afişeazâ valoarea variabilei "a"}
WriteLn(a:2);                {Se afişează valoarea variabilei "a" începînd cu
                              coloana 2}
b:=111;                      {Se atribuie valoarea 111, variabilei "b"}
WriteLn;WriteLn(a,b);        {Se afişează valorile variabilelor "a" şi "b"
                              una dupa alta, de apare "5111"}
WriteLn;WriteLn('a = ',a);   {Se afişează şirul de caractere "a = ",urmat de
                              valoarea variabilei "a"}
WriteLn;WriteLn('a = ',a,' ','b = ',b); {Se afişează două şiruri de caractere
                              şi valorile celor două variabile, despărţite
                              printr-un caracter "spaţiu"}
WriteLn;
end.
```

Citirea datelor

Citirea datelor se face începînd din poziţia curentă a cursorului şi lasă cursorul după ultimul caracter citit.

Procedurile Read şi ReadLn

Sintaxa : Read (variabila1, variabila2, …, variabila n)

Numărul minim de parametri este unu. Datele citite de la tastatură,se transferă într-o zonă tampon de memorie (bufer).

Program Citire_Scriere_Prg_0002;

```
VAR                                  {Declarare variabile}
Raza : Integer;                      {Raza cercului}

Begin                                {Începutul programului}
WriteLn;                                   {Afişează un rînd gol}
WriteLn('Introduceti raza cercului');   {Afişează textul dintre apostroafe}
ReadLn(Raza);WriteLn('Raza = ',Raza);  {Citeşte valoarea introdusă de la
                                        tastatură pentru variabila "raza" şi
                                        apoi tipăreşte această valoare}
WriteLn('Aria     cercului este ',3.14*Raza*Raza:10:3); {Tipăreşte textul
                                        dintre apostroafe şi apoi valoarea
                                        rezultată, prin calculare directă, începînd cu
                                        coloana 10, spre stînga}
WriteLn('Lungimea cercului este ',2*3.14*Raza:10:3); {La fel deasupra}
WriteLn;                                   {Afişează un rînd gol}
End.                                 {Sfirşitul programului}
```

Unit-ul Crt

Unit-ul standard **Crt** (prescurtare de la **C**athode **R**ay **T**ube) implementează un număr de subprograme axate spre aplicaţii în care ecranul este utilizat în mod caracter. Programele care utilizează aceste subprograme, trebuie să conţină directiva **uses Crt** .

Subprogramele unit-ului **Crt** pot fi clasificate astfel **:**

- De interes general
- Subprograme destinate gestiunii ferestrelor
- Subprograme orientate spre culori
- Subprograme destinate generatorului de sunet şi a intensitaţii video

În accepţiunea unit-ului **Crt**, valorile **“x”** sau **coloanele** sînt numerotate crescător, de la stînga la dreapta, iar valorile **“y”** sau **liniile** sînt numerotate crescător, de sus în jos. Un ecran cu coordonate normale are **80** de **coloane** şi **25** de **linii.**

Citiri speciale de la tastatură

Funcţia KeyPressed

Funcţia **“KeyPressed”** returnează valoarea **“True”** dacă pe claviatură s-a apăsat o tastă sau **“False”** în caz contrar. Funcţia nu sesizează apăsarea tastelor **“Shift, Alt, NumLock”**. Se foloseşte în situaţiile în care se rulează o secvenţă de instrucţiuni pînă cînd este apăsată o tastă. În cazul în care această funcţie este reapelată, trebuie golit **buffer-ul**, utilizînd funcţia **“ReadLn”** fără parametri, altfel funcţia returnează **“True”** fără ca vreo tastă să fie apăsată. Programul următor (Prg_0003) aşteaptă apăsarea unei taste pentru a se opri.

Program Functia_KeyPressed_Prg_0003;

```
Uses Crt;

Begin                    {Începe programul}
ClrScr;                  {Şterge ecranul}
Repeat                       {Repetă secvenţa de instrucţiuni
                         cuprinsă între "Repeat" şi "Until"
                         cît timp "KeyPressed" este "True"}
Write('Turbo Pascal');Until KeyPressed; {Afişează textul dintre apostroafe}
End.
```

Funcţia ReadKey

Funcţia **"ReadKey"** este de tip caracter, fără parametri formali şi returnează un caracter care s-a apăsat la tastatură. Caracterul tastat nu apare la ecran. Dacă în timpul funcţionării unui program, se ajunge la această instrucţiune, programul aşteaptă un timp nedeterminat, pînă se apasă o tastă.

Program ReadKey_Prg_0004;

```
Uses Crt;

Var
c:Char;

Begin
ClrScr;                          {Ştergerea ecranului}
c:=ReadKey;                      {Citirea caracterului tastat}
If c=#0 Then c:=ReadKey;
Case c Of                        {Instrucţiune de selecţie multiplă}
   #72:Write('Am apasat sageata sus');
   #80:Write('Am apasat sageata jos');
   #75:Write('Am apasat sageata stinga');
   #77:Write('Am apasat sageata dreapta');
Else Write('Nu am apasat tastele cu sageti');
End
End.
```

Program Prg_0005_ReadKey;

```
Uses Crt;
Var
c:Char;                          {Variabila "c" este de tip caracter}
Begin
ClrScr;                          {Şterge ecranul}
c:=ReadKey;                      {Aşteaptă să se apese o tastă}
Write('Ai apasat o tasta');      {Afişează textul dintre apostroafe}
End.
```

Ferestre ecran

O fereastră reprezintă o porţiune dreptunghiulară din ecran, în care se fac citiri şi scrieri de date. Dacă nu se declară nici o fereastră, atunci tot ecranul este fereastra implicită. În momentul în care avem declarate mai multe ferestre, numai una poate fi activă la un moment dat.

Deschiderea unei ferestre se face cu procedura **:**

Window(C_1,L_1,C_2,L_2)

Perechile **(C_1,L_1)** şi **(C_2,L_2)** reprezintă coordonatele colţurilor din stînga sus şi dreapta jos. **C_1** şi **C_2** reprezintă coloanele iar **L_1** şi **L_2** reprezintă liniile. Colţul din stînga sus al unei ferestre are coordonatele **(1,1)**. Nu există o procedură prin care se închide o fereastră. Prin deschiderea unei ferestre, se închide fereastra anterioară, adică nu mai este activă. Pentru a activa din nou o fereastră, se folosesc iar procedurile **Window, TextBackGround** şi **TextColor.**

Despre culori

Culorile sînt obţinute prin amestecul a trei culori **: R**ed (Roşu), **G**reen (Verde) şi **B**lue (Albastru). Ele sînt codificate cu numere de la 0 la 15 sau cu constante simbolice reprezentînd denumirile în engleză a culorilor.

Unit-ul **Crt** lucrează în mod text. Ecranul văzut de noi dispune de 25 de linii şi 80 de coloane. Un caracter se afişează la intersecţia dintre o linie şi o coloană. În **memoria video** pentru fiecare caracter se utilizează doi octeţi, unul pentru a reţine **codul caracterului** şi unul pentru a reţine **atributele grafice**, adică **culoarea fondului, culoarea caracterului** şi **pîlpîirea.**

Spaţiul de afişare a unui caracter are o culoare de fond **(F)**, caracterul are o culoare de afişare **(C)** şi afişarea este continuă sau intermitentă **(B = Blink).** Informaţiile referitoare la atributele caracterului sînt reţinute într-un octet. Structura acestuia pe biţi este următoarea **: BFFFCCCC.** Cei patru biţi notaţi cu **C** reţin culoarea de afişare a caracterului**.** De aici rezultă că avem la dispoziţie 2^4 culori adică **16**, codificate de la **0** la **15**. Culoarea de fond se reprezintă pe 3 biţi notaţi cu **F**, deci aem la dispoziţie 2^3 culori, adică 8, notate de la **0** la **7**. Bitul **B** indică afişarea continuă (valoarea **0**) sau intermitentă (valoarea **1**).

Culorile		Valoare	Intensitate	Roşu	Verde	Albastru
Black	Negru	0	0	0	0	0
Blue	Albastru	1	0	0	0	1
Green	Verde	2	0	0	1	0
Cyan	Turcoaz	3	0	0	1	1
Red	Roşu	4	0	1	0	0
Magenta	Mov	5	0	1	0	1
Brown	Maro	6	0	1	1	0
LightGray	Gri Deschis	7	0	1	1	1
DarkGray	Gri Închis	8	1	0	0	0
LightBlue	AlbastruDeschis	9	1	0	0	1
LightGreen	VerdeDeschis	10	1	0	1	0
LightCyan	Albast.Desc.Str	11	1	0	1	1
LightRed	RoşuDeschis	12	1	1	0	0
LightMagenta	MovDeschis	13	1	1	0	1
Yellow	Galben	14	1	1	1	0
White	Alb	15	1	1	1	1

Procedura TextBackGround(culoare)

Această procedură are un singur parametru formal şi anume culoarea. Culoarea poate fi dată în două moduri, şi anume **:** printr-o cifră de la **0** la **7** sau printr-un cuvînt în engleză, din tabelul de mai sus. Această formă de adresare a culorii este posibilă, deoarece în unit-ul **Crt**, numele culorilor sînt declarate drept constante cu valorile codurilor respective. Totuşi simpla ei apelare nu schimbă culoarea fondului. Culoarea fondului devine vizibilă în două cazuri **:**

- După rularea procedurii ClrScr
- După scrierea efectivă în fereastră

Există posibilitatea ca fiecare text scris într-o fereastră să fie identificat prin propriile culori de fond şi de scriere a caracterelor. Din acest motiv, după execuţia acestei proceduri, fereastra activă nu capătă culoarea adresată.

Procedura ClearScreen (ClrScr)

Această procedură nu are parametri şi are următoarele efete **:**

- Curăţă fereastra prin umplere cu spaţii goale (blancuri)
- Atribuie întregii ferestre culoarea fondului declarată cu **"TextBackGround"**
- Poziţionează cursorul în colţul din stînga sus.

Procedura TextColor(culoare)

Parametrul trebuie să conţină un număr cuprins între **0** şi **15** sau un cuvînt care exprimă această culoare, din tabelul de mai sus. Prin adăugarea constantei **"Blink = 128"** la culoarea scrisului, afişarea caracterelor este clipitoare.

Program Prg_0006_Ferestre;

```
Uses Crt;
Var
c:Char;                    {Se declară variabila C pentru
                            aşteptare}

Begin                      {Începutul programului}
   TextBackGround(0);      {Culoarea fondului este negru}
   ClrScr;                 {Se şterge ecranul}
   Window(1,1,40,10);      {Coordonatele ferestrei}
   TextBackGround(1);      {Culoarea fondului este albastru}
   ClrScr;                 {Se şterge ecranul}
   TextColor(14);          {Culoarea scrisului este galben}
```

```
    Write(' Fereastra nr. 1');          {Se afişează textul dintre apostroafe}
    C:=ReadKey;                         {Se aşteaptă apăsarea unei taste}

    Window(41,1,80,10);                 {Coordonatele ferestrei}
    TextBackGround(2);                  {Culoarea fondului este verde}
    ClrScr;                             {Se şterge ecranul}
    TextColor(4);                       {Culoarea scrisului este roşu}
    Write(' Fereastra nr. 2');          {Se afişează textul dintre apostroafe}
    C:=ReadKey;                         {Se aşteaptă apăsarea unei taste}

    Window(1,11,80,25);                 {Coordonatele ferestrei}
    TextBackGround(3);                  {Culoarea fondului este albastru deschis}
    ClrScr;                             {Se şterge ecranul}
    TextColor(14);                      {Culoarea scrisului este galben}
    Write(' Fereastra nr. 3');          {Se afişează textul dintre apostroafe}
    C:=ReadKey;                         {Se aşteaptă apăsarea unei taste}
End.                                    {Sfirşitul programului}
```

Program Prg_0007_Culorile_Fondului_si_Caracterelor;

```
Uses Crt;

Begin
TextBackGround(0);TextColor(15);
ClrScr;
TextBackGround(0);TextColor(15);
WriteLn(' Cod Fond = 0 (Black)       - Cod Culoare Caracter = 15 (White)   ');
TextBackGround(1);TextColor(14);
WriteLn(' Cod Fond = 1 (Blue)        - Cod Culoare Caracter = 14 (Yellow)   ');
TextBackGround(2);TextColor(13);
WriteLn(' Cod Fond = 2 (Green)   - Cod Culoare Caracter = 13 (LightMagenta)   ');
TextBackGround(3);TextColor(12);
WriteLn(' Cod Fond = 3 (Cyan)        - Cod Culoare Caracter = 12 (LightRed)   ');
TextBackGround(4);TextColor(11);
WriteLn(' Cod Fond = 4 (Red)         - Cod Culoare Caracter = 11 (LightCyan)   ');
TextBackGround(5);TextColor(10);
WriteLn(' Cod Fond = 5 (Magenta) - Cod Culoare Caracter = 10 (LightGreen)   ');
TextBackGround(6);TextColor(9);
WriteLn(' Cod Fond = 6 (Brown)       - Cod Culoare Caracter =  9 (LightBlue       ');
TextBackGround(7);TextColor(8);
WriteLn(' Cod Fond = 7 (LightGray) - Cod Culoare Caracter =  8 (DarkGray        ');
TextBackGround(0);TextColor(7);
WriteLn(' Cod Fond = 0 (Black)       - Cod Culoare Caracter =  7 (LightGray       ');
TextBackGround(1);TextColor(6);
WriteLn(' Cod Fond = 1 (Blue)        - Cod Culoare Caracter =  6 (Brown           ');
TextBackGround(2);TextColor(5);
```

```
WriteLn(' Cod Fond = 2 (Green)       - Cod Culoare Caracter =  5 (Magenta        ');
TextBackGround(3);TextColor(4);
WriteLn(' Cod Fond = 3 (Cyan)        - Cod Culoare Caracter =  4 (Red            ');
TextBackGround(4);TextColor(3);
WriteLn(' Cod Fond = 4 (Red)         - Cod Culoare Caracter =  3 (Cyan           ');
TextBackGround(5);TextColor(2);
WriteLn(' Cod Fond = 5 (Magenta)     - Cod Culoare Caracter =  2 (Green          ');
TextBackGround(6);TextColor(1);
WriteLn(' Cod Fond = 6 (Brown)       - Cod Culoare Caracter =  1 (Blue           ');
TextBackGround(7);TextColor(0);
WriteLn(' Cod Fond = 7 (LightGray)   - Cod Culoare Caracter =  0 (Black          ');
TextBackGround(11);TextColor(1+Blink);{Se adaugă la culoarea scrisului
                                       constanta "Blink" = 128}
WriteLn('Text Clipitor');

End.
```

Variabila TextAttr

Variabila **"TextAttr"** este de tip **"byte"** şi memorează atributele caracterelor. Informaţiile de culoare sînt codificate în felul următor **: BFFFCCCC**. De aceea formula de calcul a atributelor unui caracter este următoarea **:**

```
TextAttr:=CulScris+16*CulFond+Blink
```

Procedura GoToXY

Este folosită la poziţionarea cursorului în interiorul ferestrei active.

GoToXY(C,L);

C şi **L** reprezintă coordonatele relative la fereastra activă, adică coloana şi linia. În cazul coordonatelor care nu se încadrează în mărimea ferestrei, procedura nu are efect.

GoToXY(5,7); - Poziţionează cursorul pe coloana 5 din linia 7.

Procedurile WhereX şi WhereY

Aceste proceduri sînt folosite pentru a determina poziţia curentă a cursorului.

WhereX – returnează abcisa cursorului curent, adică numărul coloanei.

WhereY – returnează ordonata cursorului curent, adică numărul liniei.

Program Prg_0008_GoToXY_si_Where;

```
Uses Crt;
Begin
ClrScr;                              {Se şterge ecranul}
GoToXY(10,5);                        {Se mută cursorul pe col. 10 şi Lin.5}
Write('   Coloana 10,Linia 5');      {Se afişează textul dintre apostroafe}
WriteLn;                             {Se afişează un rînd gol}
WhereX;WhereY;                       {Se află coordonatele X şi Y ale
                                      cursorului}
WriteLn(WhereX);WriteLn(WhereY);     {Se afişează coordonatele cursorului}
End.
```

Procedura ClrEol

Pornind de la poziţia actuală a cursorului, procedura **"ClrEol"** şterge toate caracterele pînă la sfîrşitul liniei. Poziţia cursorului nu este modificată.

Program Prg_0009_ClrEol;

```
Uses Crt;
Begin
ClrScr;                              {Se şterge ecranul}
Write('Turbo Pascal 7.0');           {Se afişează textul dintre apostroafe}
ReadKey;                             {Se aşteaptă apăsarea unei taste}
GoToXY(18,1);                        {Se mută cursorul pe col. 18 si Lin. 1}
Write('este un limbaj de programare'); {Se afişează textul dintre apostroafe}
ReadKey;                             {Se aşteaptă apăsarea unei taste}
GoToXy(18,1);                        {Se mută cursorul pe col. 18 si Lin. 1}
ReadKey;                             {Se aşteaptă apăsarea unei taste}
ClrEol;                              {Se şterg caracterele de la dreapta
                                      cursorului}
ReadKey;                             {Se aşteaptă apăsarea unei taste}
End.
```

Procedura InsLine

Procedura **"InsLine"** inserează o linie goală la poziţia cursorului. Liniile situate dedesuptul cursorului vor defila în jos cu o linie şi ultima linie iese de pe ecran.

Program Prg_0010_InsLine;

```
Uses Crt;
Begin
ClrScr;                          {Se şterge ecranul}
WriteLn;                         {Se afişează un rînd gol}
WriteLn('Turbo Pascal 7.0');     {Se afişează textul dintre apostroafe}
GoToXY(1,1);                     {Se mută cursorul pe col. 1 si Lin. 1}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
InsLine;                         {Se inserează un rînd gol}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
InsLine;                         {Se inserează un rînd gol}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
InsLine;                         {Se inserează un rînd gol}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
End.
```

Procedura DelLine

Procedura **“DelLine”** şterge toată linia la care se află cursorul. Liniile situate dedesuptul cursorului, vor defila în sus cu o linie. Linia aflată la bază, va fi vidă.

Program Prg_0011_DelLine;

```
Uses Crt;
Begin
ClrScr;                          {Se sterge ecranul}
GoToXY(1,25);                    {Se mută cursorul pe col. 1 şi Lin. 25}
WriteLn('Turbo Pascal 7.0');     {Se afişează textul dintre apostroafe}
GoToXY(1,1);                     {Se mută cursorul pe col. 1 şi Lin. 1}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
DelLine;                         {Se inserează un rînd gol}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
DelLine;                         {Se inserează un rînd gol}
ReadKey;                         {Se aşteaptă apăsarea unei taste}
```

```
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
DelLine;                    {Se inserează un rînd gol}
ReadKey;                    {Se aşteaptă apăsarea unei taste}
End.
```

Variabila CheckBreak

Variabila predefinită **CheckBreak** este de tip boolean şi validează sau invalidează testarea întreruperii **Ctrl-Break**. Dacă variabila este iniţialiată cu **true**, atunci prin apăsarea simultană a tastelor **Ctrl-Break** se opreşte programul. Dacă variabila este iniţializată cu **false**, atunci apăsarea simultană a tastelor **Ctrl-Break** nu are nici un efect. Valoarea implicită a variabilei este **true**.
Exemplu **: CheckBreak:=True;**

Variabila CheckEof

Variabila predefinită **CheckEof** este de tip boolean şi validează sau invalidează caracterul de sfîrşit de fişier. Dacă variabila este iniţializată cu **true**, atunci prin apăsarea simultană a tastelor **Ctrl-Z**, se generează un caracter sfîrşit de fişier în timpul citirii dintr-un fişier asignat ecranului. Dacă variabila este iniţializată cu **false**, apăsarea simultană a tastelor **Ctrl-Z** nu are nici un efect. Valoarea implicită a variabilei este **true**.
Exemplu **: CheckEof:=True;**

Variabila DirectVideo

Variabila predefinită **DirectVideo** este de tip boolean şi validează sau invalidează accesul direct la ecran pentru operaţiile **Write** şi **WriteLn** care afişează pe ecran. Dacă variabila este iniţializată cu **true**, procedurile **Write** şi **WriteLn**, la fişierele asociate cu **Crt**, vor memora caracterele direct în memoria video, şi nu vor apela **BIOS-ul** pentru această afişare. Dacă variabila este iniţializată cu **false**, toate caracterele vor fi scrise prin apelul **BIOS-ului** şi deci procesul va fi mult mai lent.