

# **Architektura komputerów**

**Temat: Program odczytujący i wyświetlający na ekranie plik bmp w rozdzielczości 320\*200 i paletcie 256 kolorów.**

## Format BMP

Plik .bmp składa się z kilku części: nagłówek pliku bitmapy, nagłówek informacyjnego bitmapy, palety kolorów i danych obrazu. Poniższy schemat przedstawia budowę pliku .bmp.

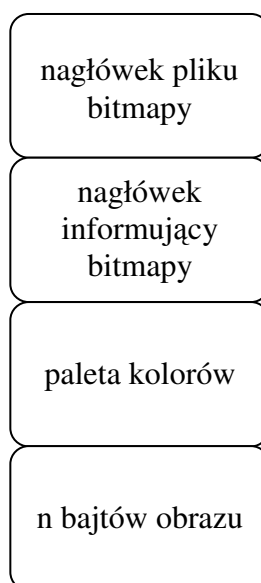


Tabela zawiera opis pól nagłówka bitmapy oraz pól nagłówka informacyjnego bitmapy

Przesunięcie wskaźnika pozycji pliku	Rozmiar pola w bajtach	Opis pola
<b>Nagłówek pliku bitmapy (14 bajtów)</b>		
00h	2	Specyfikator typu pliku są to dwa znaki określające typ przechowywanego przez bitmapę obrazu:
		BM – BitMapa (bitmapa)
		BA – Bitmap Array (tablica bitmapy)
		CI – Color Icon (kolorowa ikona)
		CP – Color Pointer (kursor myszy)
		IC – Icon (ikona)
		PT – Pointer (kursor myszy)
02h	4	Rozmiar całego pliku podany w bajtach
06h	2	Zarezerwowane, zawsze 0
08h	2	Zarezerwowane, zawsze 0
0Ah	4	Odległość od początku pliku do pierwszego bajtu obrazu, podana w bajtach

Przesunięcie wskaźnika pozycji pliku	Rozmiar pola w bajtach	Opis pola
<b>Nagłówek informacyjny bitmapy (40 bajtów)</b>		
0Eh	4	Rozmiar nagłówka informacyjnego bitmapy podany w bajtach. Dla bitmap tworzonych w różnych systemach może przybierać różną wartość
		12 – dla systemu OS/2 1.x
		40 – dla systemu Windows 3.x
		64 – dla systemu OS/2 2.x
12h	4	Szerokość obrazu podana w pikselach
16h	4	Wysokość obrazu podana w pikselach
1Ah	2	Liczba planów w obrazie, zawsze 1
1Ch	2	Liczba bitów używanych do reprezentacji jednego piksela
		1 – bitmapa zawiera monochromatyczny obraz składający się z wyłącznie z czarnych i białych pikseli
		4 – bitmapa składa się z 16 pikseli
		8 – bitmapa składa się z 256 kolorowych pikseli
		24 – bitmapa składa się z pikseli mogących przybrać jeden z 16777216 kolorów, plik bitmapy nie ma palety kolorów, każde kolejne trzy bajty obrazu oznaczają intensywność barwy czerwonej, zielonej i niebieskiej dla kolejnych pikseli
1Eh	4	Typ kompresji obrazu:
		0 – BI_RGB, obraz jest nieskompresowany
		1 – BI_RLE8, 8 bitowa kompresja RLE
		2 – BI_RLE4, 4 bitowa kompresja RLE
22h	4	Rozmiar obrazu podany w bajtach
26h	4	Pozioma rozdzielczość DPI obrazu
2Ah	4	Pionowa rozdzielczość DPI obrazu
2Eh	4	Jeżeli liczba używanych kolorów wynosi zero, bitmapa używa liczby kolorów zgodnej z liczbą bitów używanych do reprezentacji jednego piksela; jeżeli pole zawiera wartość niezerową, określa ona liczbę wzorców koloru używanych do wyświetlania obrazu
32h	4	Liczba wzorców koloru branych pod uwagę podczas ustalania kolorów pikseli; jeżeli to pole zawiera 0, to znaczy, że wszystkie kolory są używane

Jeżeli bitmapa nie przechowuje 24-bitowego obrazu, to bezpośrednio za 54 bajtami nagłówka pliku i nagłówka informacyjnego bitmapy umieszczona jest paleta kolorów; gdy natomiast na każdy piksel przypadają 24 bity, to bitmapa nie ma palety kolorów.

Paleta kolorów składa się z 256 wzorców koloru; każdy z nich możemy opisać za pomocą następującego rekordu:

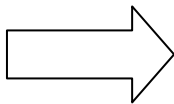
```
type wzorzec_koloru_BMP = record
    niebieski : byte;
    zielony : byte;
    czerwony : byte;
    zarezerwowany : byte;
end;
```

Zauważmy, że pola intensywności koloru czerwonego, zielonego i niebieskiego zapisane są w bitmapie w odwrotnej kolejności niż w paletce kolorów karty graficznej VGA. Poza tym wzorce oddzielone są jednobajtowymi komórkami, zarezerwowanymi dla potrzeb pliku bitmapy. Obszar, jaki zajmuje paleta kolorów w pliku bitmapy, to 1024 bajty (wynik działania  $256*4$ ).

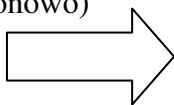
Za obszarem palety kolorów znajdują się już tylko dane obrazu; ich odczyt uzależniony jest od rodzaju kompresji obrazu. Jeżeli bitmapa zawiera nieskompresowany obraz, to możemy go odczytać sekwencyjnie, bajt po bajcie, aż do końca pliku.

Zwróćmy uwagę, że obraz w bitmapie bez względu na to, czy jest skompresowany, czy nie, zapisany jest w postaci odwróconej:

Obraz przed zapisem



Obraz po zapisie  
(odbity pionowo)



Wyjaśnienia tej sprawy należy szukać w przeszłości. Doszło wówczas do sporu między grafikami a matematykami; pierwsi chcieli, aby obraz był zapisany w normalny sposób, drudzy, aby wszystko było zgodne z matematycznym punktem widzenia. Wojnę wygrali matematycy i jest to jedyny powód, dla którego musimy dzisiaj tak się męczyć.

Aby sobie z tym problemem poradzić, wystarczy odwrócić położenie odczytanych bajtów.

## Odczytywanie obrazu z pliku BMP

Omawiany format plików .bmp jest zgodny ze specyfikacją systemu Windows; dla takich bitmap rozmiar nagłówka pliku bitmapy wynosi 14 bajtów, a rozmiar nagłówka informacyjnego bitmapy do 40 bajtów, co daje łącznie 54 bajty. Zatem tworzymy rekord, który będzie zawierał wszystkie pola wymienionych nagłówków.

```
type naglowek_pliku_BMP = record
    typ_obrazu : array[0..1] of char;
    rozmiar_pliku : longint;
    zarezerwowane1 : word;
    zarezerwowane2 : word;
    odleglosc_do_obrazu : longint;
    rozmiar_naglowka_info : longint;
    szerokosc_obrazu : longint;
    wysokosc_obrazu : longint;
    liczba_planow_obrazu : word;
    liczba_bitow_na_piksel : word;
    typ_kompresji : longint;
    rozmiar_obrazu : longint;
    pozioma_rozdziel_DPI : longint;
    pionowa_rozdziel_DPI : longint;
    liczba_uzywanych_kolorow : longint;
    liczba_znaczacych_kolorow : longint;
end;
```

Możemy teraz zdefiniować zmienną typu *naglowek\_pliku\_BMP* i odczytać do niej pierwsze 54 bajty z pliku .bmp. Spowoduje to umieszczenie w polach zmiennej wszystkich wartości pól obydwu nagłówków bitmapy, dzięki czemu w bardzo prosty sposób będziemy mogli sprawdzić jej własności. Posługiwać będziemy się wyłącznie bitmapami 256-kolorowymi, dlatego aby wyświetlane obrazy miały odpowiednie kolory, musimy odczytać również z pliku .bmp paletę kolorów. Znajduje się ona za nagłówkiem informacyjnym bitmapy; do jej odczytu używać będziemy 256-elementowej tablicy typu *wzorzec\_koloru\_BMP*, który jest następującym rekordem:

```
type wzorzec_koloru_BMP = record
    niebieski : byte;
    zielony : byte;
    czerwony : byte;
    zarezerwowany : byte;
end;
```

Obraz pliku .bmp jest odwrócony pionowo, czyli pierwsza linia jest na miejscu ostatniej, a ostatnia linia na miejscu pierwszej, druga na miejscu przedostatniej, a przedostatnia na miejscu drugiej itd. Odczytując pierwsze 320 bajtów danych obrazu, odczytujemy całą jego ostatnią linię; odczytane bajty musimy umieścić w pamięci między adresem pierwszego bajtu ostatniej linii a adresem jej ostatniego bajtu, czyli pomiędzy adresem A000:F8C0 a adresem A000:F9FF (F8C0 dziesiętnie wynosi 63680, a F9FF równe jest dziesiętnie 63999).

Kod źródłowy procedury takiego programu będzie następujący:

```
Procedure wyswietlenie (NazwaPlikuP:String;opoznienie:word);
var
plik : file;
licznik, wynik : word;
licznik_petli : word;
naglowek : naglowek_pliku_BMP;
paleta_k : array[0..255] of wzorzec_koloru_BMP;

begin
clrscr;
If (rozmiar(NazwaPlikuP)=false) and (opoznienie=0) then
Begin
    Bład;
    ClrSCR;
    Writeln('Rozmiar lub paleta kolorow podanego obrazu jest nieprawidlowa.');
```

```
    Writeln('Nacisnij ENTER aby wyjsc do glownego menu.');
```

```
    Readln;
End else
If (rozmiar(NazwaPlikuP)=false) and (opoznienie<>0) then else
Begin

assign(plik, NazwaPlikuP);

reset(plik,1);

licznik := sizeof(naglowek_pliku_BMP);

blockread(plik,naglowek, licznik, wynik);
If opoznienie=0 then
    Begin
        writeln('Typ obrazu : ', naglowek.typ_obrazu[0], naglowek.typ_obrazu[1]);
        writeln('Rozmiar pliku : ', naglowek.rozmiar_pliku);
        writeln('Zarezerwowane1 : ', naglowek.zarezerwowane1);
        writeln('Zarezerwowane2 : ', naglowek.zarezerwowane2);
        writeln('Odleglosc do obrazu : ',naglowek.odleglosc_do_obrazu);
        writeln('Rozmiar naglowka info. : ', naglowek.rozmiar_naglowka_info);
        writeln('Szerokosc obrazu : ', naglowek.szerokosc_obrazu);
        writeln('Wysokosc obrazu : ', naglowek.wysokosc_obrazu);
        writeln('Liczba planow obrazu : ', naglowek.liczba_planow_obrazu);
        writeln('Liczba bitow na piksel : ', naglowek.liczba_bitow_na_piksel);
        writeln('Typ kompresji : ', naglowek.typ_kompresji);
        writeln('Rozmiar obrazu : ', naglowek.rozmiar_obrazu);
        writeln('Pozioma rozdziel. DPI : ', naglowek.pozioma_rozdziel_DPI);
        writeln('Pionowa rozdziel. DPI : ', naglowek.pionowa_rozdziel_DPI);
        writeln('Uzywanych kolorow : ', naglowek.liczba_uzywanych_kolorow);
        writeln('Znaczacych kolorow : ', naglowek.liczba_znaczacych_kolorow);
        writeln('');
        writeln('Wcisnij dowolny klawisz aby kontynuowac');
```

```
        readkey;
    End;
End;
```

```
licznik := sizeof(paleta_k);

blockread(plik, paleta_k, licznik, wynik);

asm {wlaczenie trybu 13h}
    mov ah, 00h    {zaladuj 00h do AH - numer funkcji przerwania 10h}
    mov al, 13h   {zaladuj 13h do AL - numer trybu pracy karty graficznej}
    int 10h       {wywolaj przerwanie 10h}
end;

port[$3c8] := 0; {zaczynamy zapis palety kolorow od pierwszego jej wzorca}

for licznik_petli := 0 to 255 do {zapis w petli wszystkich 255 wzorcow}
begin
port[$3c9] := paleta_k[licznik_petli].czerwony shr 2;
port[$3c9] := paleta_k[licznik_petli].zielony shr 2;
port[$3c9] := paleta_k[licznik_petli].niebieski shr 2;
end; {koniec petli for}

licznik := 320;

for licznik_petli := 0 to 199 do
blockread(plik, mem[$A000:(199 - licznik_petli)*320], licznik, wynik);

close(plik);
Delay(opoznienie);
If opoznienie=0 then
Begin
readkey;
    asm {wyjscie z trybu 13h}
        mov ah, 00h    {zaladuj 00h do AH - numer funkcji przerwania 10h}
        mov al, 03h   {zaladuj 13h do AL - numer trybu pracy karty graficznej}
        int 10h       {wywolaj przerwanie 10h}
    end;
end;
end;
End;
```

*{poniżej została omówiona zasada działania procedury. Objasnienie dotyczy samego wyświetlania obrazu}*

Na początku w procedurze następuje definicja wcześniej omówionych typów danych, nagłówek\_pliku\_BMP i wzorzec\_koloru\_BMP. Następny krok to definicja zmiennych, liczników, zmiennych plikowych oraz zmiennych do przechowywania nagłówka bitmapy i jej palety kolorów.

Procedura rozpoczyna działanie od wyczyszczenia ekranu w trybie tekstowym, kojarzy zmienną plikową ze zbiorem danych *nazwa\_podanego\_pliku.bmp*, otwiera go, ustalając jednocześnie liczbę bajtów odczytywanych podczas każdego cyklu odczytu na 1. Następnie do zmiennej *licznik* zapisujemy rozmiar typu *naglowek\_pliku\_BMP*, co jest równoznaczne z ustaleniem liczby cykli odczytu na 54. Kolejna czynność to odczytanie do zmiennej *naglowek* pierwszych 54 bajtów z pliku bitmapy. Po odczytaniu pół nagłówków program informuje nas o ich zawartości, wyświetlając serię komunikatów. Po naciśnięciu dowolnego klawisza następuje ustalenie liczby cykli odczytów na równą rozmiarowi zmiennej *tablica\_k* (czyli 1024) i odczytanie palety kolorów bitmapy. Kolejny krok to uruchomienie trybu graficznego 13h i załadowanie odczytanej palety kolorów do palety kolorów karty graficznej. Zauważmy, że paletę kolorów ładujemy dopiero po uruchomieniu trybu 13h. Podczas każdej zmiany trybu pracy karty graficznej zostaje załadowana domyślna paleta kolorów, która niszczy poprzednią. Zauważmy również, że wartości intensywności kolorów przed zapisem do rejestru 3c9h są dzielone przez cztery. Większość programów służących do edycji 256-kolorowych obrazów tworzy

paletę kolorów, w której pola intensywności podstawowych barw przyjmują wartości z zakresu od 0 do 255: taka paleta jest zapisywana w plikach bitmapy.

Natomiast pola intensywności podstawowych barw palety kolorów karty graficznej mogą przyjmować jedynie wartości z zakresu od 0 do 63. Należy więc przekonwertować liczbę z zakresu od 0 do 255 do liczby z zakresu od 0 do 63. Najprostrzym sposobem na to jest podzielenie liczby przez 4, co jest równoznaczne przesunięciu jej bitów o dwie pozycje w prawo.

Po załadowaniu palety kolorów ustalamy liczbę cykli odczytów na 320; oznacza to, że podczas każdego wykonania pętli `for` procedura `BlockRead` odczyta 320 razy po jednym bajcie z pliku *nazwa\_podanego\_pliku.bmp*. Obraz wyświetlany jest linia po linii, począwszy od dołu ekranu. Procedura po wyświetleniu obrazu czeka na dowolny klawisz, po którego naciśnięciu kończy działanie.

Cały program posiada jeszcze kilka interesujących opcji. W prostym menu, w którym, poprzez naciśnięcie odpowiedniego klawisza, możemy:

- zmienić ścieżkę katalogu, w którym znajdują się pliki bmp
- listing plików w katalogu (nie tylko bmp)
- wyświetlić plik bmp
- usunąć plik
- uruchomić slideshow

Kod źródłowy programu:

```
Program ACDSsee;
uses Dos, Crt;
type wzorzec_koloru_BMP = record
    niebieski : byte;
    zielony : byte;
    czerwony : byte;
    zarezerwowany : byte;
end;

type naglowek_pliku_BMP = record
    typ_obrazu : array[0..1] of char;
    rozmiar_pliku : longint;
    zarezerwowane1 : word;
    zarezerwowane2 : word;
    odleglosc_do_obrazu : longint;
    rozmiar_naglowka_info : longint;
    szerokosc_obrazu : longint;
    wysokosc_obrazu : longint;
    liczba_planow_obrazu : word;
    liczba_bitow_na_piksel : word;
    typ_kompresji : longint;
    rozmiar_obrazu : longint;
    pozioma_rozdzziel_DPI : longint;
    pionowa_rozdzziel_DPI : longint;
    liczba_uzywanych_kolorow : longint;
    liczba_znaczacych_kolorow : longint;
end;

var Sciezka, NazwaPliku:string;
    opoznienie:word;

Procedure Blad;
Begin
    Sound(300);
    Delay(100);
    Nosound;
End;
```

```
Function rozmiar(NazwaPlikuP:string):boolean;
var naglowek:naglowek_pliku_bmp;
    a,licznik,wynik:word;
    plik:file;
Begin
    rozmiar:=true;
    assign(plik, NazwaPlikuP);
    reset(plik,1);
    licznik := sizeof(naglowek_pliku_BMP);
    blockread(plik,naglowek, licznik, wynik);
    If (naglowek.rozmiar_pliku<>65078) then rozmiar:=false;
    Close(plik);
End;

Procedure wyswietlenie(NazwaPlikuP:String;opoznienie:word);
var
plik : file;
licznik, wynik : word;
licznik_petli : word;
naglowek : naglowek_pliku_BMP;
paleta_k : array[0..255] of wzorzec_koloru_BMP;

begin
clrscr;
If (rozmiar(NazwaPlikuP)=false) and (opoznienie=0) then
Begin
    Blad;
    ClrSCr;
    Writeln('Rozmiar lub paleta kolorow podanego obrazu jest nieprawidlowa.');
```

```
    Writeln('Nacisnij ENTER aby wyjsc do glownego menu.');
```

```
    Readln;
End else
If (rozmiar(NazwaPlikuP)=false) and (opoznienie<>0) then else
Begin

assign(plik, NazwaPlikuP);

reset(plik,1);

licznik := sizeof(naglowek_pliku_BMP);

blockread(plik,naglowek, licznik, wynik);
If opoznienie=0 then
    Begin
        writeln('Typ obrazu : ', naglowek.typ_obrazu[0], naglowek.typ_obrazu[1]);
        writeln('Rozmiar pliku : ', naglowek.rozmiar_pliku);
        writeln('Zarezerwowane1 : ', naglowek.zarezerwowane1);
        writeln('Zarezerwowane2 : ', naglowek.zarezerwowane2);
        writeln('Odleglosc do obrazu : ',naglowek.odleglosc_do_obrazu);
        writeln('Rozmiar naglowka info. : ', naglowek.rozmiar_naglowka_info);
        writeln('Szerokosc obrazu : ', naglowek.szerokosc_obrazu);
        writeln('Wysokosc obrazu : ', naglowek.wysokosc_obrazu);
        writeln('Liczba planow obrazu : ', naglowek.liczba_planow_obrazu);
        writeln('Liczba bitow na piksel : ', naglowek.liczba_bitow_na_piksel);
        writeln('Typ kompresji : ', naglowek.typ_kompresji);
        writeln('Rozmiar obrazu : ', naglowek.rozmiar_obrazu);
        writeln('Pozioma rozdziel. DPI : ', naglowek.pozioma_rozdziel_DPI);
        writeln('Pionowa rozdziel. DPI : ', naglowek.pionowa_rozdziel_DPI);
        writeln('Uzywanych kolorow : ', naglowek.liczba_uzywanych_kolorow);
        writeln('Znaczaczych kolorow : ', naglowek.liczba_znaczaczych_kolorow);
        writeln('');
        writeln('Wcisnij dowolny klawisz aby kontynuowac');
```

```
        readkey;
    End;
End;
```

```
licznik := sizeof(paleta_k);

blockread(plik, paleta_k, licznik, wynik);

asm {wlaczenie trybu 13h}
    mov ah, 00h    {zaladuj 00h do AH - numer funkcji przerwania 10h}
    mov al, 13h   {zaladuj 13h do AL - numer trybu pracy karty graficznej}
    int 10h       {wywolaj przerwanie 10h}
end;

port[$3c8] := 0; {zaczynamy zapis palety kolorow od pierwszego jej wzorca}

for licznik_petli := 0 to 255 do {zapis w petli wszystkich 255 wzorcow}
begin
port[$3c9] := paleta_k[licznik_petli].czerwony shr 2;
port[$3c9] := paleta_k[licznik_petli].zielony shr 2;
port[$3c9] := paleta_k[licznik_petli].niebieski shr 2;
end; {koniec petli for}

licznik := 320;

for licznik_petli := 0 to 199 do
blockread(plik, mem[$A000:(199 - licznik_petli)*320], licznik, wynik);

close(plik);
Delay(opoznienie);
If opoznienie=0 then
Begin
readkey;
    asm {wyjscie z trybu 13h}
        mov ah, 00h    {zaladuj 00h do AH - numer funkcji przerwania 10h}
        mov al, 03h   {zaladuj 13h do AL - numer trybu pracy karty graficznej}
        int 10h       {wywolaj przerwanie 10h}
    end;
end;
End;
end;

Procedure Otwieranie_Blad(var NazwaPlikuP:string);
    var klawisz:char;
    pdu:file;
    Begin
        Writeln;
        Assign(pdu,NazwaPlikuP);
        {$I-}
        Reset(pdu);
        While IOResult<>0 do
        Begin
            ClrScr;
            Blad;
            Writeln('Podales zla nazwe pliku. ');
            Writeln('Podaj nazwe jeszcze raz (q-wyjscie)');
            Readln(NazwaPliku);
            If NazwaPliku='q' then break else
            Begin
                Assign(pdu,NazwaPliku);
                Reset(pdu);
            End;
        End;
    End;
    {$I+}
End;
```

```
Procedure listing(sciezkaP,NazwaPlikuP:string);
var DirInfo:SearchRec;
Begin
  ChDir(sciezkaP);
  ClrScr;
  If NazwaPlikuP='b' then NazwaPlikuP:= '*.bmp';
  Writeln('Oto pliki ',NazwaPlikuP,' z katalogu ',sciezkaP);
  Writeln;
  FindFirst(NazwaPlikuP, 0, DirInfo);
  while DosError = 0 do
  begin
    Writeln(DirInfo.Name);
    FindNext(DirInfo);
  end;
  Writeln;
  Writeln('Nacisnij dowolny klawisz');
  Repeat
  Until keypressed;

End;

Procedure ZmianaKatalogu(var sciezkaP:string);
var AktualnyKatalog:String;
    klawisz:char;
Begin
  Writeln;
  ClrScr;

  GetDir(0,AktualnyKatalog);
  Writeln('Obecnie jestes w katalogu ',AktualnyKatalog);
  Writeln;
  Writeln('Czy chcesz zmienic katalog? (t-tak/n-nie)');
  Repeat
  klawisz:=Uppcase(readkey);
  Until (klawisz='T') or (klawisz='N');
  Writeln;
  If klawisz='T' then Begin
    Writeln;
    Writeln('Podaj sciezke dostepu:');
    Writeln;
    Readln(sciezkaP);
    {$I-}
    ChDir(sciezkaP);
    While IOResult<>0 do
    Begin
      ClrScr;
      Blad;
      Writeln('Podales nieprawidlowa
sciezke. ');
      Writeln('Podaj sciezke jeszcze raz (q-
wyjscie) ');
      Readln(sciezkaP);
      If sciezkaP<>'q' then ChDir(sciezkaP);
    End;
    {$I+}
  End;
End;

End;
```

```
Procedure UP; {usuniecie pliku}
var pdu:file; {plik do usuniecia}
    klawisz:char;
Begin
    ClrScr;
    Writeln('Czy chcesz usunac plik (t-tak/n-nie)?');
    Repeat
        klawisz:=Ucase(readkey);
    Until (klawisz=Ucase('t')) or (klawisz=Ucase('n'));
    If klawisz='T' then
        Begin
            Writeln;
            Writeln('Podaj nazwe pliku do usuniecia:');
            Readln(NazwaPliku);
            Otwieranie_blad(NazwaPliku);
            Assign(pdu,NazwaPliku);
            If NazwaPliku<>'q' then
                Erase(pdu);
        End;
    {$I+}

End;

Procedure SlideShow;
var DirInfo:SearchRec;
Begin
    ClrScr;
    Writeln('Podaj wartosc opoznienia wyswietlania plikow (w
milisekundach):');
    {$I-}
    Readln(opoznienie);
    While IOResult<>0 do
        Begin
            ClrScr;
            Blad;
            Writeln('Podales zla wartosc opoznienia. ');
            Writeln('Podaj ja jeszcze raz:');
            Readln(opoznienie)
        End;
    {$I+}
    FindFirst('*.*bmp', 0, DirInfo);
    If DirInfo.size<>65078 then Begin
        Blad;
        ClrScr;
        Writeln('W podanym katalogu nie ma
plikow *.*bmp');
        Writeln('Nacisnij ENTER aby powrocic do
glownego menu. ');
        Readln;
    End
    else
        Begin
            while DosError = 0 do
                begin
                    wyswietlenie(DirInfo.name,opoznienie);
                    FindNext(DirInfo);
                end;
        asm {wyjscie z trybu 13h}
            mov ah, 00h {zaladuj 00h do AH - numer funkcji przerwania 10h}
            mov al, 03h {zaladuj 13h do AL - numer trybu pracy karty graficznej}
            int 10h      {wywolaj przerwanie 10h}
        end;
    end;
End;
```

```
Procedure x; {licencja}
Begin
  ClrScr;
  Writeln('Zlamanie warunkow licencjonowania programu grozi
sformatowaniem dysku ');
  Writeln('i calkowitym rozmrozeniem jedzenia w zamrazarce.');
```

Writeln('Pelna tresc licencji znajduje sie na stronie  
www.instaltrojan.lol');

```
  Readln;
  exit;
End;
```

```
Procedure Glowny;
Begin
  lowvideo;
  Writeln('Witamy w najnowszej wersji przegladarki EjSiDiSi.');
```

Writeln;

```
  Writeln('Ze wzgledu na to, ze ten komputer jest zbyt slaby, ');
  Writeln('mozliwe jest tylko przegladanie plikow graficznych w
formacie bmp.');
```

Writeln('(w rozdzielczosci 320\*200 i 256 kolorach)');

```
  Writeln;
  Writeln('Nacisnij ENTER aby przejsc do menu.');
```

textcolor(black);

```
  Readln;

End;
```

```
Procedure CoChceszZrobic;
var klawisz:char;
Begin
  Repeat

    Writeln;
    ClrScr;
    textcolor(white);
    lowvideo;
    Writeln('Z-Zmiana katalogu');
```

Writeln('L-Listing plikow w katalogu');

```
    Writeln('W-Wyświetlanie pliku bmp');
```

Writeln('U-Usuwanie pliku');

```
    Writeln('S-SlideShow');
```

Writeln('X-Warunki licencji');

```
    Writeln('Q-wyjście');
```

gotoxy(45,23);

```
    Writeln('Program wykonał Wojciech Gasiewski');
```

Repeat

```
      klawisz:=Uppcase(readkey);
    Until (klawisz=Uppcase('z')) Or (klawisz=Uppcase('x')) Or
(klawisz=Uppcase('s')) Or (klawisz=Uppcase('w')) Or
      (klawisz=Uppcase('l')) Or (klawisz=Uppcase('u')) Or
(klawisz=Uppcase('q'));
```

Case klawisz of

```
  'Z':ZmianaKatalogu(Sciezka);
  'L':Begin
    ClrScr;
    Writeln('Podaj nazwe pliku do wyszukania. Mozesz uzyc
symboli "*" i "?');
```

Writeln('Jesli jako nazwe pliku podasz literke "b" to  
wyswietlone zostana pliki \*.bmp');

```
    Readln(NazwaPliku);
    Listing(Sciezka,NazwaPliku);

  End;
  'W':Begin
    Writeln;
    ClrScr;
```

```
Writeln('Podaj nazwe pliku (bez rozszerzenia; zostanie ono
automatycznie dopisane):');
Readln(NazwaPliku);
Nazwapliku:=NazwaPliku+'.bmp';
Otwieranie_blad(NazwaPliku);
If NazwaPliku<>'q' then
Begin
    opoznienie:=0;
    wyswietlenie(NazwaPliku,opoznienie);
End;
End;
'S':SlideShow;
'U':UP;
'X':x;
End;
Until klawisz=Ucase('q');
End;
```

Procedure **Matrix**;

var i,j,a,b,c:byte;

Begin

clrscr;

Randomize;

highvideo;

Delay(500);

Write('F');

Delay(120);

Write('o');

Delay(120);

Write('l');

Delay(120);

Write('l');

Delay(120);

Write('o');

Delay(120);

Write('w');

Delay(120);

Write(' ');

Delay(120);

Write('t');

Delay(120);

Write('h');

Delay(120);

Write('e');

Delay(120);

Write(' ');

Delay(120);

Write('w');

Delay(120);

Write('h');

Delay(120);

Write('i');

Delay(120);

Write('t');

Delay(120);

Write('e');

Delay(120);

Write(' ');

Delay(120);

Write('r');

Delay(120);

Write('a');

Delay(120);

Write('b');

Delay(120);

Write('b');

```
Delay(120);
Write('i');
Delay(120);
Write('t');
Delay(620);
lowvideo;
ClrScr;
Textcolor(green);
Delay(500);
  for i:=1 to 22 do begin

    a:=Random(10);
    b:=Random(10);
    c:=Random(10);

    If i>=4 then begin
      gotoxy(1,i-3); {1}
      writeln(a); end;

    If i>=10 then begin
      HighVideo;
      gotoxy(3,i-9); {3}
      writeln(b);
      LowVideo; end;

    If i>=15 then begin
      gotoxy(5,i-14); {5}
      writeln(c); end;

    If i>=6 then begin
      gotoxy(7,i-5); {7}
      writeln(a); end;

    If i>=10 then begin
      gotoxy(9,i-19); {9}
      writeln(b); end;

    If i>=1 then begin
      HighVideo;
      gotoxy(11,i); {11}
      writeln(c);
      LowVideo; end;

    If i>=7 then begin
      gotoxy(13,i-6); {13}
      writeln(a); end;

    If i>=8 then begin
      HighVideo;
      gotoxy(15,i-7); {15}
      writeln(b);
      LowVideo; end;

    If i>=15 then begin
      gotoxy(17,i-14); {17}
      writeln(c); end;

    If i>=3 then begin
      gotoxy(19,i-2); {19}
      writeln(a); end;

    If i>=1 then begin
      HighVideo;
      gotoxy(21,i); {21}
      writeln(b);
```

```
LowVideo; end;

If i>=5 then begin
gotoxy(23,i-4);    {23}
writeln(c); end;

If i>=2 then begin
gotoxy(25,i-1);    {25}
writeln(a); end;

If i>=1 then begin
gotoxy(27,i);      {27}
writeln(b); end;

If i>=9 then begin
HighVideo;
gotoxy(29,i-8);    {29}
writeln(c);
LowVideo; end;

If i>=11 then begin
gotoxy(31,i-10);   {31}
writeln(a); end;

If i>=5 then begin
HighVideo;
gotoxy(33,i-4);    {33}
writeln(b);
LowVideo; end;

If i>=4 then begin
gotoxy(35,i-3);    {35}
writeln(c); end;

If i>=1 then begin
gotoxy(37,i);      {37}
writeln(a); end;

If i>=5 then begin
HighVideo;
gotoxy(39,i-4);    {39}
writeln(b);
LowVideo; end;

If i>=1 then begin
gotoxy(41,i);      {41}
writeln(c); end;

If i>=6 then begin
gotoxy(43,i-5);    {43}
writeln(a); end;

If i>=10 then begin
gotoxy(45,i-9);    {45}
writeln(b); end;

If i>=4 then begin
HighVideo;
gotoxy(47,i-3);    {47}
writeln(c);
LowVideo; end;

If i>=8 then begin
gotoxy(49,i-7);    {49}
writeln(a); end;
```

```
If i>=2 then begin
HighVideo;
gotoxy(51,i-1);    {51}
writeln(b);
LowVideo; end;
```

```
If i>=6 then begin
gotoxy(53,i-5);    {53}
writeln(c); end;
```

```
If i>=1 then begin
HighVideo;
gotoxy(55,i);      {55}
writeln(a);
LowVideo; end;
```

```
If i>=3 then begin
gotoxy(57,i-2);    {57}
writeln(b); end;
```

```
If i>=7 then begin
gotoxy(59,i-6);    {59}
writeln(c); end;
```

```
If i>=2 then begin
gotoxy(61,i-1);    {61}
writeln(a); end;
```

```
If i>=5 then begin
HighVideo;
gotoxy(63,i-4);    {63}
writeln(b);
LowVideo; end;
```

```
If i>=1 then begin
gotoxy(65,i);      {65}
writeln(c); end;
```

```
If i>=6 then begin
gotoxy(67,i-5);    {67}
writeln(a); end;
```

```
If i>=9 then begin
gotoxy(69,i-8);    {69}
writeln(b); end;
```

```
If i>=7 then begin
HighVideo;
gotoxy(71,i-6);    {71}
writeln(c);
LowVideo; end;
```

```
If i>=1 then begin
gotoxy(73,i);      {73}
writeln(a); end;
```

```
If i>=4 then begin
gotoxy(75,i-3);    {75}
writeln(b); end;
```

```
If i>=3 then begin
gotoxy(77,i-2);    {77}
writeln(c); end;
```

```
    If i>=6 then begin
    HighVideo;
    gotoxy(79,i-5);    {79}
    writeln(a);
    LowVideo; end;

    Delay(80);end;
Normvideo;
END;

Begin
asm {wyjscie z trybu 13h}
    mov ah, 00h {zaladuj 00h do AH - numer funkcji przerwania 10h}
    mov al, 03h {zaladuj 13h do AL - numer trybu pracy karty graficznej}
    int 10h    {wywolaj przerwanie 10h}
end;

ClrScr;
Glowny;
CoChceszZrobic;
Matrix;
End.
```