Pakiet `crt` zawiera procedury i funkcje, które pozwalają na sterowanie trybem ekranu, kolorami, oknami i dźwiękiem. Opisy procedur są zawarte w pliku pomocy, można je wyświetlić z menu `Help – Standard units – Crt – CRT Functions and Procedures`. Wybrane zmienne, funkcje i procedury zdefiniowane w module `crt` przedstawiono w tabeli 1.

Tabela 1. Wybrane zmienne, funkcje i procedury zdefiniowane w module `crt`.

Nazwa	Opis
`WindMin`	Zmienna typu `word` przechowująca współrzędne lewego górnego narożnika ekranu (`x` w pierwszym słowie, `y` w drugim).
`WindMax`	Zmienna typu `word` przechowująca współrzędne prawego dolnego, narożnika ekranu (`x` w pierwszym słowie, `y` w drugim).
`WhereX`	Funkcja typu `byte` zwracająca wartość aktualnej współrzędnej `x` kursora.
`WhereY`	Funkcja typu `byte` zwracająca wartość aktualnej współrzędnej `y` kursora.
`KeyPressed`	naciśnięty dowolny klawisz, a `false` w przeciwnym wypadku. Aby usunąć kod klawisza z bufora klawiatury należy go przeczytać, np. funkcja `ReadKey`.
`ReadKey`	Funkcja zwracająca z bufora klawiatury kod pierwszego klawisza.
`ClrScr`	Procedura czyści ekran i ustawia kursor w lewym górnym rogu ekranu.
`GotoXY(x, y: byte)`	Procedura pozwala na umieszczenie kursora w miejscu o współrzędnych `x`, `y` bieżącego okna. Domyślnie oknem jest cały ekran, a więc zakres współrzędnych to `[1..80]` dla współrzędnej poziomej `x` i `[1..25]` dla współrzędnej pionowej `y`.
`DelLine`	Procedura czyści zawartość wiersza, w którym znajduje się kursor.
`Delay(ms: word)`	Procedura przerywa działanie programu na `ms` milisekund.
`Sound(Hz: word)`	Procedura rozpoczyna emitowanie przez głośnik komputera dźwięku o częstotliwości `Hz` (aby zakończyć należy wywołać procedurę `NoSound`).
`NoSound`	Procedura powodująca zakończenie wydawania dźwięku.
`TextColor(kolor: byte)`	Procedura, która ustala nowy kolor wyświetlania znaków na ekranie. Wartość koloru tekstu można podać w postaci stałej symbolicznej zdefiniowanej w module `crt` (nazwy kolorów po angielsku) lub liczby w zakresie `0÷15`.
`TextBackground(kolor: byte)`	Procedura ustalająca kolor tła dla wpisywanego tekstu. Wartość koloru tła można podać w postaci stałej symbolicznej zdefiniowanej w module `crt` (nazwy kolorów po angielsku) lub liczby w zakresie `0÷7`.

1. Napisz program, który wczyta tekst, wyczyści ekran i wyświetli tekst w ośmiu predefiniowanych kolorach.

```
program p1;
uses crt;
var tekst: string;
    i    : byte;
begin
  clrscr;
  repeat
    write('Podaj tekst: ');
```

```
    readln(tekst);
  until tekst <> '';
  clrscr;
  for i := 0 to 7 do
  begin
    textcolor(i);
    writeln(tekst);
  end;
  readln;
end.
```

2. Zmodyfikuj program tak, aby wprowadzony tekst był wyświetlany we wszystkich kolorach znaku i w kilku wybranych kolorach tła.

3. Napisz program, który wstrzymuje swoje działanie na 5 sekund, (w tym czasie będziesz wciskał(a) dowolne klawisze na klawiaturze), a po zakończeniu przerwy wyświetli znaki zgromadzone w buforze klawiatury.

```
program p2;
uses crt;
var tekst: string;
begin
  clrscr;
  writeln('Czekam 5 sekund, w tym czasie wciskaj dowolne klawisze');
  delay(5000);
    {a}
  writeln('Koniec czekania');
  write('Teraz podaj liczbe: ');
  readln(tekst);
  writeln('Wprowadziles tekst: ', tekst);
  readln;
end.
```

4. Zmodyfikuj wcześniejszy program tak, aby na bieżąco był czyszczony bufor klawiatury. W tym celu zadeklaruj dodatkową zmienną `znak: char` **oraz w wierszu oznaczonym przez** `{a}` **wpisz polecenie:** `while keypressed do znak := readkey;`

Uruchom program i zinterpretuj wyniki.

5. Napisz program generujący dźwięki o stałej częstotliwości z malejącym stopniowo odstępem czasowym między kolejnymi dźwiękami.

```
program p3;
uses crt;
var i: Integer;
begin
  for i := 20 downto 1 do
  begin
    sound(500);
    delay(40);
    nosound;
    delay(5*i)
  end;
end.
```

6. Zmodyfikuj wcześniejszy program tak, aby kolejne dźwięki miały coraz wyższą częstotliwość.

7. Wykonaj zadania 1 i 4.

W poprzednich ćwiczeniach ekran monitora znajdował się w trybie tekstowym. Był on traktowany jak pole złożone z prostokątnych pozycji znakowych. W każdej pozycji można było umieścić jeden znak. Jednak ekran można również traktować jak pole złożone z pojedynczych punktów (pikseli).

Procedury i funkcje graficzne są zdefiniowane w module `graph`. Dlatego każdy program, który realizuje operacje w trybie graficznym, musi mieć deklarację tego modułu. Wybrane funkcje i procedury przedstawiono w tabeli 2.

Tabela 2. Wybrane funkcje i procedury zdefiniowane w module `graph`.

Nazwa	Opis
`GetMaxX`	Funkcja podająca maksymalną dozwoloną wartość współrzędnej `x`.
`GetMaxY`	Funkcja podająca maksymalną dozwoloną wartość współrzędnej `y`.
`GetMaxColor`	Funkcja podająca najwyższy dozwolony kolor. Najczęściej jest to kolor biały. Rysowanie w tym kolorze zapewnia działanie na każdym komputerze.
`InitGraph(karta_graficzna, nr_trybu, ścieżka_do_sterownika)`	Procedura do włączania trybu graficznego.
`CloseGraph`	Procedura kończąca pracę w trybie graficznym nie ma parametrów). Po jej wywołaniu następuje powrót do trybu tekstowego.
`DetectGraph(sterownik,karta)`	Procedura sprawdzająca sprzęt i możliwości sterownika.
`SetColor(numer koloru)`	Procedura ustala obowiązujący kolor rysowania dla większości procedur. Od momentu użycia procedury wszystko będzie rysowane we wskazanym kolorze (poza figurami które określają swój kolor za pomocą procedury `PutPixel`).
`SetBKColor(numer koloru)`	Procedura określająca kolor tła.
`SetFillStyle(wzór, kolor)`	Procedura ustalająca kolor i wzór do wypełnienia zamkniętego konturu.
`PutPixel(x, y, numer_koloru)`	Procedura rysuje punkt na określonych współrzędnych `x` i `y` w określonvm kolorze.
`Line(xp, yp, xk, yk)`	współrzędnymi `xp, yp` do punktu określonego współrzędnymi `xk, yk` w kolorze ustalonym przez ostatnią procedurę. Nie zmienia położenia kursora graficznego.
`LineTo(x, y)`	Procedura rysuje odcinek od bieżącego położenia kursora graficznego to wskazanego punktu i przenosi kursor graficzny do punktu końcowego. Kursora graficznego nie widać na ekranie.
`MoveTo(x,y)`	Procedura przenosi kursor graficzny do punktu o współrzędnych `(x, y)`, nie rysuje linii.
`Circle(x, y, r)`	`r`. Kolor linii jest taki jaki został określony w ostatniej procedurze `SetColor`.
`Rectangle(x1, y1, x2, y2)`	Procedura rysuje prostokąt, gdzie `x1, y1` – współrzędne lewego górnego rogu, `x2, y2` – prawego dolnego rogu. Kolor linii jest taki jaki został określony w ostatniej procedurze `SetColor`.
`Bar(x1, y1, x2, y2)`	Procedura rysuje prostokąt i wypełnia go kolorem ustalonym wcześniej w procedurze `SetFillStyle`.
`FloodFill(x, y, kolor brzegu)`	`SetFillStyle` pole, które zawiera punkt o współrzędnych `x, y` i jest ograniczone konturem narysowanym w kolorze okreslonym przez kolor brzegu.

8. Podstawowe informacje o pracy w trybie graficznym.

Do włączania trybu graficznego służy procedura:

`InitGraph(karta_graficzna, nr_trybu, ścieżka_do_sterownika)`

Znaczenie poszczególnych argumentów jest następujące:

`karta_graficzna` – parametr przekazywany przez zmienną (nie może być wyrażeniem). Określa kartę graficzną komputera. Aby program mógł być uruchomiony na każdym komputerze, powinien sam zbadać, z jakiej karty graficznej będzie korzystać. Aby tak się stało należy przypisać zmiennej `karta_graficzna` wartość zero, symbolicznie oznaczoną słowem `detect`.

`nr_trybu` – zmienna określająca sposób pracy karty graficznej. Jeżeli używamy wartości `detect` dla karty, to numer trybu nie jest istotny, komputer sam wybierze tryb pracy karty (zmienna musi jednak wystąpić w wywołaniu procedury).

`ścieżka_do_sterownika` – wyrażenie typu `string` określające ścieżkę dostępu do sterownika karty graficznej (program zapisany w pliku z rozszerzeniem `bgi`). Każda karta ma własny program `bgi` i nasz program nie będzie mógł wykonywać operacji graficznych, jeżeli na dysku nie znajdzie odpowiedniego pliku `bgi`. Sterowniki kart są najczęściej zapisane w katalogu `..\TP\BGI`.

Po wykonaniu procedury `InitGraph` komputer jest gotowy do rysowania, cały ekran jest czarny. Nie wskazane jest wówczas używanie procedur `read, readln, write, writeln`.

Do wyłączania trybu graficznego służy procedura: `CloseGraph`.

Każdy punkt na ekranie ma współrzędne określone przez liczby typu `word` i nazwane literami `X` i `Y`. Współrzędna `X` rośnie w prawo, `Y` rośnie do dołu. Lewy górny róg ma współrzędne `(0,0)`. To ile jest punktów zależy od karty grafiki i trybu graficznego.

Punkt kreślony na ekranie powinien mieć określone położenie i kolor. Liczba dostępnych kolorów zmienia się w zależności od rodzaju zainstalowanej karty i sposobu pracy monitora. Kolory są ponumerowane od `0`. Kolor czarny ma zawsze numer `0`, a biały `1`. Po włączeniu trybu graficznego wszystkie punkty mają kolor czarny.

9. Napisz program kreślący na ekranie monitora 10 linii o losowej długości i położeniu.

```
program p4;
uses crt, graph;
var karta, tryb, MaxX, MaxY, i, x1, x2, y1, y2: Integer;
    znak: char;
begin
  karta := detect;
  InitGraph(karta, tryb, 'c:\tp\bgi');
  MaxX := GetMaxX;
  MaxY := GetMaxY;
  randomize;               {Uruchomienie generatora liczb losowych}
  for i := 1 to 10 do
  begin
    x1 := random(MaxX);    {Funkcja losująca liczbę całkowitą}
    x2 := random(MaxX);    {z przedziału [0, zakres)         }
    y1 := random(MaxY);
    y2 := random(MaxY);
    Line(x1, y1, x2, y2)
  end;
  ReadKey;
  CloseGraph;
end.
```

10. Wykonaj zadania 6 i 7.

Zadania

1. Napisz program wygaszacza ekranu, który wyświetla gwiazdki w dowolnych miejscach ekranu do momentu naciśnięcia dowolnego klawisza.

2. Zmodyfikuj program wygaszacza tak, aby wyświetlał dowolne znaki, w różnych kolorach, na różnych tłach.

3. Zmodyfikuj program wygaszacza tak, aby działał on do czasu wprowadzenia hasła.

4. Napisz program, który generuje dźwięk o dowolnej częstotliwości i czasie trwania. Po uruchomieniu program powinien generować prostą melodię.

5. Napisz program, który wyświetli na całym ekranie 1000 punktów losowych we wszystkich dostępnych kolorach.

6. Napisz program, który wyświetli na środku ekranu koło w kolorze czerwonym a w nim trójkąt w kolorze żółtym.

7. Napisz program, który wyświetli na ekranie cztery takie same prostokąty, umieszczone symetrycznie w czterech rogach ekranu.