Dokumentation der benötigten Klassen

`import pm.gamewindow.*;`	Importiert das GameWindow-Package
`import java.awt.*;`	Importiert das AWT-Package (für die Klasse Color)

Klasse GameWindow

Konstruktor (Beispiel): `window = new GameWindow(50, 50, 800, 600, "Beispiel");` Erzeugt ein GameWindow an der Bildschirm-Position x=50 y=50 mit Breite 800 und Höhe 600.	
`void clear()`	Übermalt das Fenster in weiß.
`void clear(Color pc)`	Übermalt das Fenster mit der im Parameter pc gegebenen Farbe.
`void drawLine(int px1, int py1, int px2, int py2, Color pc)` Zeichnet eine Linie von px1 / py1 nach px2 / py2 in der Farbe pc.	
`void paintFrame()`	Überträgt das bisher gezeichnete an den Bildschirm.

Klasse Color

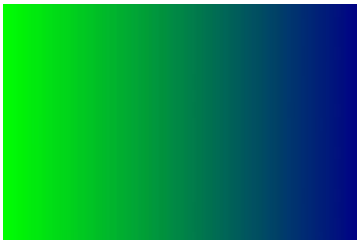
Konstruktor (Beispiel): farbe `= new Color(0.0f, 0.5f, 1.0f);` Erzeugt ein Color-Objekt mit den Anteilen R(ot) = 0, G(rün) = 50% und B(lau) = 100% Intensität. Die Parameter werden als float-Werte übergeben.
Beispiele für gängige Farben: Rot R 1 G 0 B 0 / Grün R 0 G 1 B 0 / Blau R 0 G 0 B 1 / Gelb R 1 G 1 B 0 Orange R 1 G 0,5 B 0 / Lila R 1 G 0 B 1 / Braun R 0,5 G 0,25 B 0 Weiß R 1 G 1 B 1 / Grau R 0,5 G 0,5 B 0,5 / Schwarz R 0 G 0 B 0

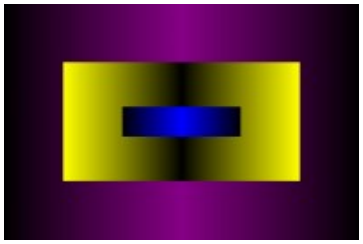
Aufgabe

Nutze für die Aufgaben die Vorlage mit der Klasse Farbverlaeufe für das Hauptobjekt.
Das Hauptobjekt hat nur ein GameWindow-Objekt.
GameImages oder Sprites werden für die Aufgabe nicht benötigt.
Color-Objekte werden nur „lokal“, also innerhalb der Methoden deklariert und erzeugt.

a) Implementiere die Methode **aufgabe1()**:
Das Fenster wird komplett mit einer Farbe, z.B. rot gefüllt. In einer while-Schleife füllt es sich immer wieder, aber die Farbe ändert sich langsam zu einer anderen, z.B. blau.

b) Implementiere die Methode **aufgabe2()**:
Das Fenster wird mit senkrechten Linien gefüllt, die gleichmäßig von einer Farbe übergehen in eine andere, z.B. von grün nach dunkelblau (oder einfacher: von schwarz nach weiß).
Tipp: rufe paintFrame() erst nach der Schleife auf.





c) Implementiere die Methode **aufgabe3()**:
Lege mehrere Verläufe übereinander, wobei die bedeckten Flächen immer kleiner werden.
Die Verläufe im Beispiel gehen dabei erst von einer Farbe zur anderen (z.B. von gelb nach schwarz) und wieder zurück, das lässt sich am einfachsten jeweils mit zwei while-Schleifen lösen.

d) Implementiere die Methode **aufgabe4()**:
Das Fenster wird wie in b) gefüllt, aber in jedem Schritt mit zwei Linien. Die unteren Linien fangen über die ganze Höhe an und werden kürzer, die oberen entsprechend länger, und die Verläufe sind entgegengesetzt.

e) **Kommentiere** deine Klasse und deine Methoden (jeweils ein Satz).

f) **Freestyle**: Überlege dir eigene Muster und Übergänge mit Farben.

Hilfestellungen

zu a)
Um von Blau nach rot zu wechseln, fängt der Blau-Anteil bei 1 an, Rot und Grün bei 0.
Blau muss dann im Verlauf der Schleife um 0.01 weniger werden, während Rot um 0.01 wächst.
Grün ändert sich nicht, bleibt bei 0.

zu b)
Nutze die Methode window.drawLine(x1, y1, x2, y2, farbe).
Die Linien sind senkrecht, die erste wird ganz links im Fenster gezeichnet, die zweite einen Pixel weiter rechts, usw. Nehmen wir an, das Fenster hat eine Breite von 800 und eine Höhe von 600:
Die erste Linie hätte also die Werte window.drawLine(0, 0, 0, 600, farbe).
Für die weiteren Linien erhöht sich x um jeweils um 1, die y-Werte bleiben gleich:
drawLine(1, 0, 1, 600, farbe), dann drawLine(2, 0, 2, 600, farbe), usw.
Um das ganze Fenster zu bedecken braucht es 800 Wiederholungen, von x = 0 bis x = 799.

Die Farben müssen dann in 800 Schritten z.B. von 0 auf 1 wachsen, d.h. 1 / 800 pro Wiederholung.
In Java kann man das (mit float-Werten) einfach so schreiben: r = r + 1.0f / 800.0f;

zu d)
Du kannst die Aufgabe mit zwei while-Schleifen lösen.
Wie in b) braucht es senkrechte Linien, jetzt ändern sich aber auch die y-Werte.
Die erste Linie ist nur ganz kurz, die zweite etwas länger, usw.
Für das obere Dreieck z.B. drawLine(0, 0, 0, 1, farbe), drawLine(1, 0, 1, 2, farbe), usw.
Allerdings ändern sich die y-Werte nicht in 1er-Schritten, da das Fenster nicht quadratisch ist.
Um in 800 Schritten von y=0 bis y=600 zu kommen, müsste sich der y-Wert in jeder Wiederholung um 600 / 800 (oder 2/3) ändern. Diese Änderungen lassen sich nicht durch Integer darstellen, der y-Wert muss also ein float- (oder double-) Wert sein.
Die Methode drawLine() erwartet dann aber wieder Integer-Parameter. Um eine float-Variable für einen Integer-Wert einzusetzen, muss man an der Stelle den float-Wert in einen Integer-Wert umwandeln (dabei werden die Nachkommastellen abgeschnitten).

```
float y;
...
while (...)
{
   ...
   y = y + 600.0f / 800.0f;    // Addiere den Bruch 600/800
   ...
   window.drawImage(..., (int)y, ...);   // setze y als int ein
   ...
}
```