Dokumentation der benötigten Klassen (Neu in blau, rot muss selbst implementiert werden)

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

Klasse GameWindow

Konstruktor (Beispiel): `window = new GameWindow(50, 50, 800, 600, "Spiel");` Erzeugt ein GameWindow an Position x=50 y=50 auf dem Bildschirm mit Breite 800 und Höhe 600.	
`void` **`clear`**`()`	Übermalt das Fenster mit weiß.
`void` **`paintFrame`**`()`	Überträgt das bisher gezeichnete an den Bildschirm
`int` **`getWidth`**`()` / `int` **`getHeight`**`()`	Geben die Breite / Höhe des Fensters zurück
`int` **`getMouseX`**`()` / `int` **`getMouseY`**`()`	Geben die aktuelle Position des Mauszeigers zurück
`boolean` **`isKeyDown`**`(int pKeyCode)` Bsp: `if (window.isKeyDown(KeyEvent.VK_RIGHT)) …` Gibt true zurück, falls die durch pKeyCode angegebene Taste gerade gedrückt ist. Für pKeyCode werden Codes der Klasse KeyEvent eingesetzt, unter anderem `VK_LEFT, VK_RIGHT, VK_UP, VK_DOWN`: Pfeiltasten `VK_A … VK_Z, VK_0 … VK_9`: Buchstaben-Tasten / Nummern 0 bis 9 `VK_SPACE, VK_ENTER`: Leertaste, Enter	

Klasse Sprite

Konstruktor (Beispiel): `sp = new Sprite("./images/ghost.png");`	
`void` **`setPosition`**`(` `  double px, double py)`	Setzt die Position des Sprites im GameWindow (bezogen auf die linke obere Ecke des Sprite-Bildes)
`void setX(double px)` `void setY(double py)`	Setzen jeweils nur die x- bzw. y-Koordinate
`void` **`setSpeed`**`(double ps)`	Setzt die Geschwindigkeit für bewege() in Pixeln
`void` **`setRichtung`**`(double pr)`	Setzt die Richtung für bewege() in Grad (0° entspr. x-Achse)
`void` **`dreheZu`**`(double x, double y)`	Dreht die Richtung zum Punkt x / y.
`void` **`draw`**`(GameWindow pwin)`	Zeichnet das Bild des Sprites an der aktuellen Position x / y auf das GameWindow pwin.
`double getX() / double getY()`	Geben die x- bzw. y-Koordinate zurück
`int getWidth() / int getHeight()`	Geben die Breite bzw. Höhe des GameImage des Sprites zurück

Klasse GameImage

Konstruktor (Beispiel): `img = new GameImage("./images/bild.png");`	
`int getWidth() / int getHeight()`	Geben die Breite bzw. Höhe des Bildes zurück

Aufgabe

Nutze für die Aufgaben die bereitgestellte Vorlage.

Klasse Sprite:

a) Implementiere die Methoden **getX()**, **getY()**, **getWidth()** und **getHeight()**.

getX() und getY() geben den Werte des Attributs x bzw. y zurück.
getWidth() und getHeight() geben die Breite bzw. Höhe des Sprite-Bildes zurück.
Diese müssen also jeweils erst vom Bild abgefragt und dann zurückgegeben werden.

Klasse Game:

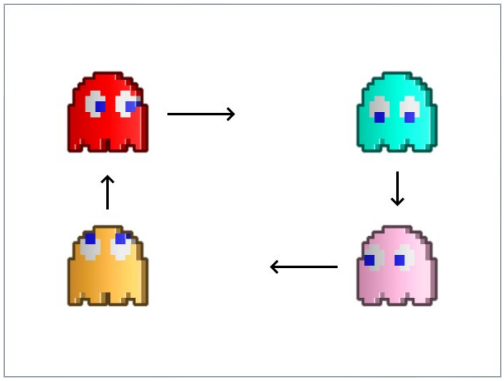
b) Implementiere die Methode **aufgabe1()**:

Einer der Sprites beginnt **genau in der Mitte des Fensters**.
Für die Berechnung braucht es Breite und Höhe des Fensters und Breite und Höhe des Sprites.

Der Sprite bewegt sich entsprechend der Pfeiltasten:
Pfeil oben → Sprite bewegt sich nach oben, Pfeil rechts → Sprite bewegt sich nach rechts usw.
Der Sprite darf das Fenster jedoch **nicht verlassen**. Er muss immer **vollständig sichtbar** sein.
Wenn er sich z.B. am rechten Rand befindet, und die Pfeiltaste rechts gedrückt wird, bewegt er sich nicht weiter nach rechts.

c) Implementiere die Methode **aufgabe2()**:

Setze die vier Sprites in die Ecken des Fensters.
Sie bewegen sich entsprechend der Grafik.
Jedesmal, wenn sie den Rand des Fensters erreichen, prallen sie quasi vom Rand ab und bewegen sich in der entgegengesetzten Richtung weiter.



d) Implementiere die Methode **aufgabe3()**:

Alle vier Sprites beginnen in der Mitte des Fensters.
Der erste Sprite folgt langsam dem Mauszeiger (d.h. bewegt sich ständig in dessen Richtung).
Der zweite folgt dem ersten, der dritte dem zweiten und der vierte dem dritten. Dabei sind die nachfolgenden Sprites immer etwas langsamer als ihre Vorgänger, so dass sie sich durch die Bewegung wie eine elastische Kette auseinanderziehen.