Prüfung 3: Processing

# Aufgabe 1: Bildbearbeitungsprogramm lesen (5 Punkte)

Sie finden in einer Sammlung von Bildbearbeitungseffekten folgendes Processing-Programm – wie so oft in der Praxis gänzlich frei von Kommentaren:

|  |
| --- |
| **public** **void** setup() {  original = loadImage("bild.jpg");  original.loadPixels();  neu = loadImage("bild.jpg");  neu.loadPixels();  **int** halbeBreite = neu.width / 2;  **int** halbeHoehe = neu.height / 2;  **for** (**int** neuY = 0; neuY < halbeHoehe; neuY ++) {  **for** (**int** neuX = 0; neuX < halbeBreite; neuX ++) {  **int** originalX = neuX + halbeBreite;  **int** originalY = neuY + halbeHoehe;  **int** originalFarbe = getColor(original, originalX, originalY);  setColor(bearbeitet, neuX, neuY, originalFarbe);  }  }  **for** (**int** neuY = halbeHoehe; neuY < bearbeitet.height; neuY ++) {  **for** (**int** neuX = halbeBreite; neuX < bearbeitet.width; neuX ++) {  **int** originalX = neuX - halbeBreite;  **int** originalY = neuY - halbeHoehe;  **int** originalFarbe = getColor(original, originalX, originalY);  setColor(bearbeitet, neuX, neuY, originalFarbe);  }  }  bearbeitet.updatePixels();  size(original.width \* 2, original.height);  } |

* Der Einfachheit halber ist oben nur die Methode setup() dargestellt. Beachten Sie, dass die gleiche Bilddatei in das Original-Bild (Variable original) und das neue Bild (Variable neu) geladen wird. **Zu Beginn sind also die beiden Bilder identisch!**
* Die Methode **getColor(bild, x, y)** liefert wie im Skript den Farbwert im Bild bild an der Koordinate (x, y).
* Die Methode **setColor(bild, x, y, farbe)** setzt den Farbwert farbe im Bild bild an Koordinate (x, y).

## Aufgabe 1.1: Programm simulieren (3 Punkte)

Um das Programm zu analysieren, spielen Sie es von Hand auf Papier durch.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Der Einfachheit halber verwendet das Bild nur schwarz (s) und weiss (w) als Farben. Das Bild ist 8x8 Pixel gross:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | | Zeichnen Sie unten das resultierende Bild hin, in gleicher Art wie das Originalbild links: Jedes Kästchen ist ein Pixel:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  | |

## Aufgabe 1.2: Programm beschreiben (2 Punkte)

Beschreiben Sie in wenigen Worten präzise den Bildbearbeitungseffekt, den das Programm darstellt: Was geschieht mit welchen Bildpunkten / Bildbereichen? Was macht die erste verschachtelte Schleife, was macht die zweite verschachtelte Schleife?

|  |
| --- |
|  |

# Aufgabe 2: Koordinaten, Formen, Interaktionen mit der Maus (5 Punkte)

Schreiben Sie ein Programm, das die Maus mit einer “Zielscheibe” verfolgt:

|  |  |
| --- | --- |
| 1. Das Programm soll einen dicken Punkt (Durchmesser 10 Pixel) darstellen, umrandet von einem Kreis (Durchmesser 30 Pixel): | 2. Wenn der Benutzer eine Maustaste drückt, sollen der Punkt und der Kreis rot dargestellt werden: |

## Aufgabe 2.1 Programm schreiben (3 Punkte)

Der Anfang ist bereits gemacht – ergänzen Sie die fehlenden Befehle, damit sich das Programm wie beschrieben verhält:

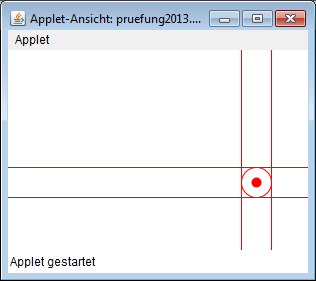
|  |
| --- |
| **public** **void** draw() {  noFill(); // keine Füllfarbe  background(255); // Hintergrund weiss, alles löschen  } |

Zur Erinnerung:

* Die Variablen mouseX, mouseY geben die aktuelle Mauskoordinate (x, y) an.
* Die Variable mousePressed wird auf true gesetzt, wenn Benutzer Maustaste drückt.
* Die Strichfarbe wird mit stroke(r, g, b) definiert.
* Die Strichstärke wird mit strokeWeight(gewicht) definiert.
* Den dicken Punkt können Sie mit point(x,y) an Koordinate (x,y) zeichnen.
* Den Kreis können Sie mit ellipse(x, y, breite, hoehe) zeichnen, wobei die Koordinaten (x, y) den Mittelpunkt festlegen und breite und hoehe den Durchmesser in horizontaler bzw. vertikaler Richtung.

## Aufgabe 2.2 Programm ergänzen (2 Punkt)

Erweitern Sie das Programm so, dass die roten „Führungslinien“ gezeichnet werden, wenn eine Maustaste gedrückt ist:



1. Markieren Sie im obigen Programm deutlich, wo Sie das Programm ergänzen müssen.  
2. Schreiben Sie anschliessend den zu ergänzenden Programmteil auf:

|  |
| --- |
|  |

Zur Erinnerung:

* Die Variablen width, height geben die Breite und Höhe der Zeichnungsfläche an.
* Eine Linie zeichnen Sie mit line(x1,y1,x2,y2), wobei die Koordinaten (x1,y1) den einen Endpunkt und die Koordinaten (x2,y2) den anderen Endpunkt definieren.