Weitere Aufgaben

Aufgabe 1:

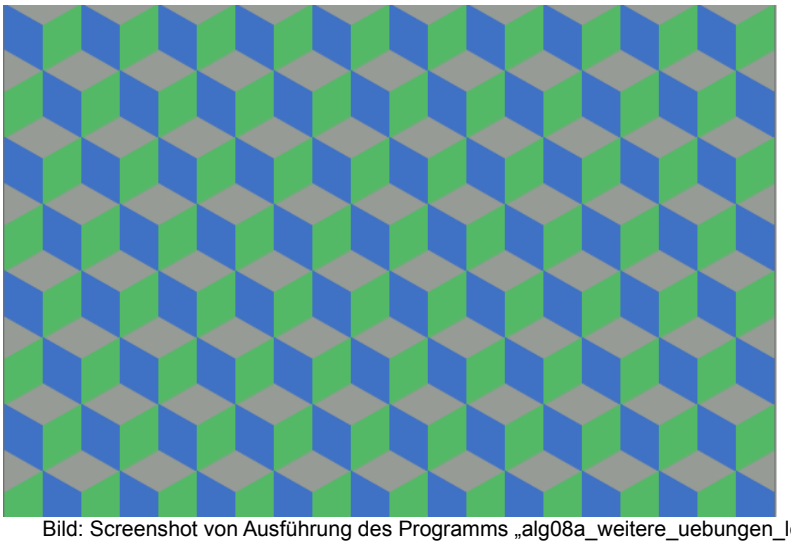


Bild: Screenshot von Ausführung des Programms „alg08a_weitere_uebungen_loesungen“ (Eisenmann)

1. *Überlege dir zunächst, aus welchen Figuren diese optische Täuschung besteht.*
2. *Schreibe eine Methode `viereck()`, der die Füllfarbe und die Koordinaten der vier Eckpunkte übergeben werden können. Nutze dazu die Methode `quad()`[1].*
3. *Das Zeichnen der Rauten einer Farbe kann man sich sparen, wenn man zu Beginn den Hintergrund in dieser Farbe füllt. Überlege dir, welche Rauten dazu am geeignetsten sind und schreibe die `setup()`-Methode mit Einstellen der Fenstergröße und Färben des Hintergrunds (in der Vorlage: grau).*
4. *Programmiere dann das Zeichnen der Rauten der beiden anderen Farben (in der Vorlage: blau und grün im Wechsel).*
5. *Verändere ggf. so, dass das ganze Fenster gefüllt wird.*

1 Siehe Blatt „Methoden zum Zeichnen“

Aufgabe 2:

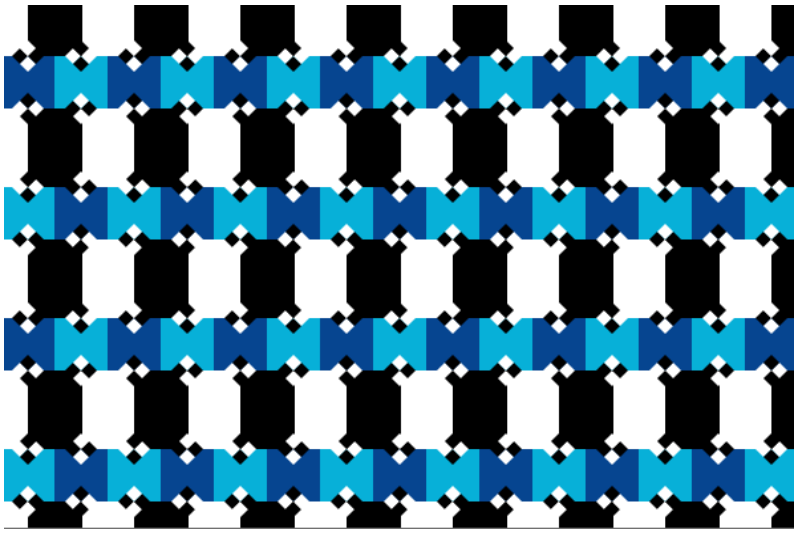


Bild: Screenshot von Ausführung des Programms „alg08b_weitere_uebungen_loesungen“ (Eisenmann)

1. *Programmiere zunächst die schwarzen, senkrechten Balken.*
2. *Programmiere die waagrechten Balken, in denen sich hell- und dunkelblaue Quadrate abwechseln.*
3. *Um die kleinen Vierecke – zusammengesetzt aus vier Rauten – zeichnen lassen zu können, brauchst du wie in Aufgabe 1 die Methode* `quad()`*. Schreibe eine Methode* `viereck()`*, der du die Farbe und alle Koordinaten der Eckpunkte übergeben kannst und dann eine Methode* `vier_rauten()`*, der eine Farbe und die Koordinaten eines Punktes übergeben werden können. Hier bietet sich der Punkt an, in dem sich alle vier Rauten treffen.*
   (Wenn du Aufgabe 1 gelöst hast, kannst du selbstverständlich die Methode `viereck()` *von dort nutzen.)*
4. *Ergänze dein Programm so, dass die optische Täuschung gezeichnet wird.*
5. *Schreibe ggf. eine zweite Version so, dass die optische Täuschung das ganze Fenster füllt.*

Aufgabe 3:

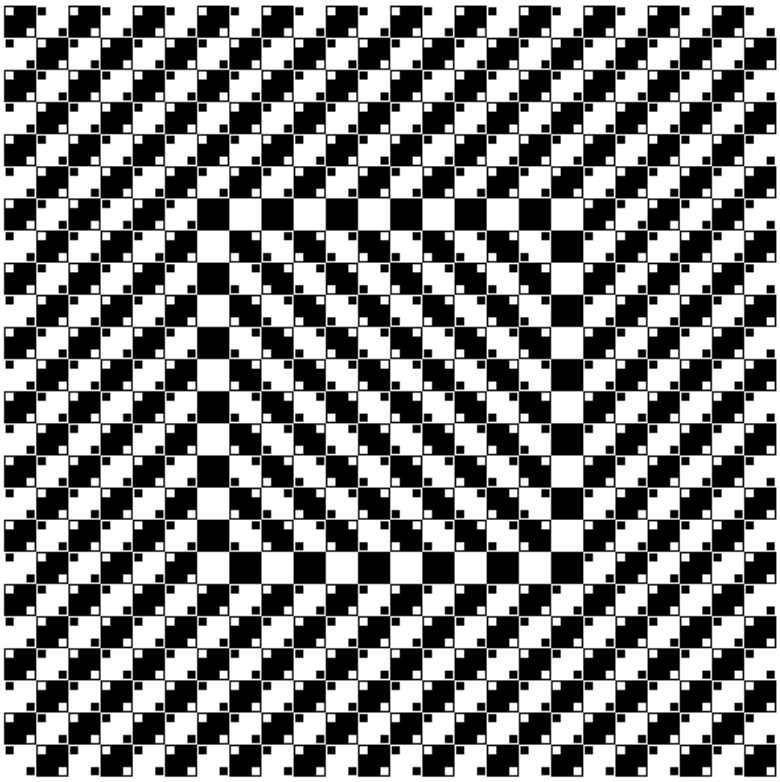


Bild: Screenshot von Ausführung des Programms „alg08c_weitere_uebungen_loesungen“ (Eisenmann)

1. *Schreibe eine Methode `zeichneQuadrat()`, der du neben der Startposition, die Grundfarbe und die Richtung der beiden kleinen Quadrate übergeben kannst.*
2. *Programmiere die optische Täuschung zuerst so, dass das ganze Feld so gefüllt wird, wie in den ersten Reihen.*
3. *Überlege dir anschließend, wie du die Mitte zeichnen lassen kannst und probiere deine Idee aus.*

Processing bietet noch viele weitere Möglichkeiten. :)

Aufgabe 4:

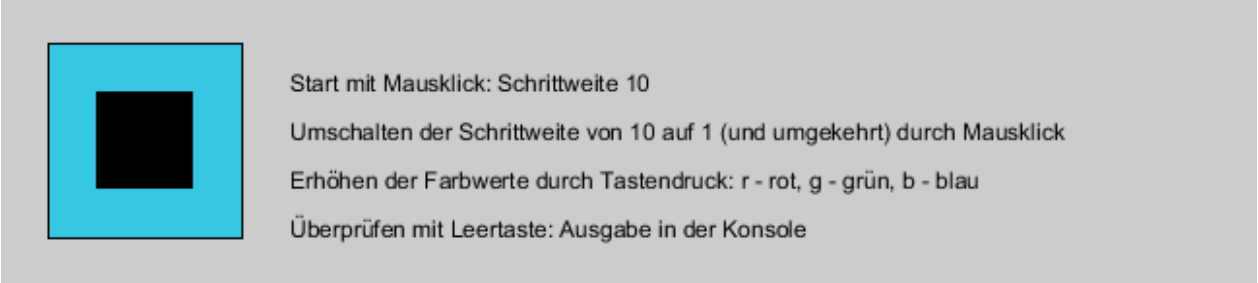


Bild: Screenshot von Ausführung des Programms „alg08d_weitere_uebungen_loesungen“ (Eisenmann)

Hier geht es um ein Farbenspiel. Das große Quadrat wird mit einer zufälligen Farbe gefärbt. Die einzelnen Farbwerte (rot, grün und blau) des inneren Quadrates kann man durch Tastaturdruck verändern (siehe Anleitung auf dem Screenshot). Mit der Leertaste (ASCII-Code: 32) wird überprüft und man bekommt die Information, ob die Farben zu viel, zu wenig oder genau richtig gewählt sind.

1. *Schreibe eine Methode `vorbereitung()`, in der du die beiden Quadrate zeichnen und den Text ausgeben lässt. Rufe diese Methode in der `setup()`-Methode auf. Nutze für die Farben jeweils zwei Variablen (z.B. `rot` und `rot_hintergrund`).*
2. *Für das Spiel brauchst du Maus- und Tastaturereignisse. Lies dir die Information unten auf diesem Blatt durch und programmiere dann Schritt für Schritt die nötigen Ereignismethoden. Denke dabei ans Testen! (Lasse dir z.B. die Farbwerte in der Konsole ausgeben: `println(…)`.)*

Möchte man in Processing **Maus- und Tastaturereignisse** nutzen, so braucht man die Methode `draw()`. Diese Methode wird ständig neu aufgerufen. Die Häufigkeit der Aufrufe gibt man in der `setup()`-Methode mit `frameRate(a)` vor. Der Wert a steht für die Anzahl der Aufrufe pro Sekunde.

Braucht man keinen ständigen Aufruf wie in unserem Programm hier in der Aufgabe, so nutzt man `noLoop()` in der `setup()`-Methode. Durch `redraw()` kann man dann aktiv die `draw()`-Methode aufrufen. In der Aufgabe bietet sich das bei jedem Tastendruck an.

Als Mausereignis brauchen wir `mousePressed()`, um zwischen den beiden Schrittweiten umzuschalten und als Tastaturereignis `keyPressed()`. Hier bekommt man bei jedem Tastendruck einen Wert in der Variable `key` übergeben. Zur Überprüfung der Buchstabentasten kann man direkt mit den Buchstaben arbeiten (z.B. `if (key == 'b')` …). Nutzt man Sondertasten, wie hier die Leertaste, so braucht man den zugehörigen ASCII-Code.

Die Mausposition bei den Mausereignissen erhält man mit `mouseX` und `mouseY`.

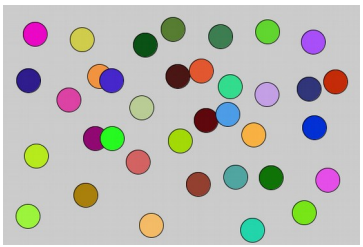
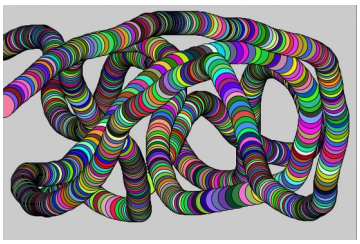
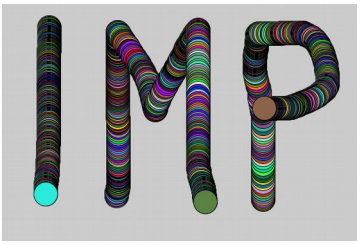
Weitere Informationen findest du in der Processing-Referenz[2] unter Input.

3. *Ergänze dein Programm nach Belieben (Versuche zählen; Möglichkeit, auch den Farbwert kleiner zu machen; …).*

2 *https://processing.org/reference/* (abgerufen am 26.04.2019)

Aufgabe 5:

Hier brauchst du die Informationen aus Aufgabe 4.

Bilder: Screenshots von Ausführung des Programms „alg08e_weitere_uebungen_loesungen“ (Eisenmann)

1. *Beim Klick auf das Fenster soll ein Kreis mit zufälliger Füllfarbe an der Mausposition gezeichnet werden. (Abbildung 1)*
2. *Wird die Maus über das Fenster bewegt (`mouseMoved()`), soll an der Mausposition ein Kreis mit zufälliger Füllfarbe gezeichnet werden. (Abbildung 2)*
3. *Nur bei gedrückter Maustaste wird gezeichnet (`mouseDragged()`). Gezeichnet wird dann wie in Aufgabe 2. (Abbildung 3)*
4. ****** Stelle Processing auf den Android-Modus um. Ändere die Größenangabe in der `setup()`-Methode auf `size(displayWidth,displayHeight)` und teste dann Nr. 3 auf dem Smartphone.*