

1.0.1. Computer können nur Nullen und Einsen abspeichern! Alle Daten (Zahlen, Text, Bilder, Musik, Videos, Programme, Computerspiele) müssen deshalb in geeignete Folgen von Nullen und Einsen umgewandelt werden. Eine Folge von Nullen und Einsen ist nichts anderes als eine als Binärzahl dargestellte natürliche Zahl.

Slogan 1.0.2.

Jede Datei bzw. Computerinformation ist eine Folge von Nullen und Einsen, also eine Binärzahl!

2. SPEICHERUNG VON BILDERN

Installiere Photoshop und Adobe Illustrator o.ä., damit du später keine Zeit verlierst.

2.1. Bilder am Computer.

2.1.1. Bildschirme bestehen meist aus einem quadratischen Raster kleiner Quadrate, den sogenannten *Pixel* (Kofferwort aus englisch *picture element*, also Bildelement), deren Farbe variiert werden kann; die einzelnen Pixel kann man je nach Auflösung des Bildschirms mit blossen Auge oder mit einer Lupe sehen.

Um ein Bild auf einem Bildschirm anzuzeigen, muss man für jedes Pixel die Farbe angeben. Ein sehr verbreitetes Modell zur Angabe (oder Codierung) der Farbe ist das sogenannte RGB-Farbmodell.

2.2. RGB-Farbmodell.

Aufgabe 2.2.1. Suche im Internet nach `#07affe`.

2.2.2. Die üblichen Suchmaschinen sollten dir eine hellblaue Farbe anzeigen. Warum?

Die Zeichenkette „07affe“ besteht aus drei zweistelligen *Hexadezimalzahlen* (was eine Hexadezimalzahl ist, wird in dieser Fussnote¹ erklärt):

$$\boxed{07affe} = \underbrace{\boxed{07}}_{\text{Rotanteil}}, \underbrace{\boxed{af}}_{\text{Grünanteil}}, \underbrace{\boxed{fe}}_{\text{Blauanteil}}$$

- 07 gibt den Rotanteil unserer Farbe an; dezimal ist dies die Zahl $0 \cdot 16 + 7 \cdot 1 = 7$;
- af gibt den Grünanteil an; dezimal ist dies die Zahl $10 \cdot 16 + 15 \cdot 1 = 175$;
- fe gibt den Blauanteil an; dezimal ist dies die Zahl $15 \cdot 16 + 14 \cdot 1 = 254$;

Zusammen mit dem vorangestellten Hash-Symbol # ist `#07affe` ein sogenannter *hex code* = *hexadecimal color code*. Beachte, dass nach dem # genau sechs Zeichen aus dem Zeichenvorrat 0, 1, 2, ..., 9, a, b, c, d, e, f folgen.

Zur Farbkodierung wird hier das RGB-Modell verwendet (RGB = **R**ed, **G**reen, **B**lue = rot, grün, blau; bitte die englischen Farbwörter merken, damit man nicht G als gelb statt grün interpretiert). Für jede der drei Farben (in der Reihenfolge RGB) speichert man den Farbanteil als eine Zahl zwischen 0 und $(255)_{10} = (\text{ff})_{16}$ (= maximaler Farbanteil), die meist als zweistellige Hexadezimalzahl angegeben wird. Für jeden der drei Farbanteile benötigt man also ein Byte (= 8-stellige Binärzahl = 2-stellige Hexadezimalzahl) und insgesamt drei Byte für die Farbe.

Die Farbe ergibt sich dann durch **additive Farbmischung** so, wie sich die Lichtkegel von rotem, grünem und blauem Scheinwerferlicht auf einer weissen Wand „addieren“. Unterscheide dies von der **subtraktiven Farbmischung**, die etwa beim Drucken verwendet wird (CMYK-Farbmodell).

Aufgabe 2.2.3. (a) Welche Farbe kodieren die *hex codes* `#ff0000`, `#00ff00`, `#0000ff`?

(b) Welche Farbe kodieren die *hex codes* `#00ffff`, `#ff00ff`, `#ffff00`?

(c) Welche Farbe kodiert `#1e53df`? Ändere den Rot-Wert so, dass die Farbe zu Lila wird. Hinweis: Nutze dazu und auch später zur Farbwahl einen „color picker“ aus dem Internet, etwa den [RGB-Farbwähler auf W3schools](#).

(d) Wie viele Farben kann man durch hex codes mit dem RGB-Modell kodieren? ²

2.3. Verfahren zur Speicherung von Bildern.

2.3.1. Die zwei wichtigsten Verfahren zur Speicherung von Bildern sind

- Rastergrafik (auch Pixelgrafik, *raster graphics*, *bitmap*, *pixmap*)
- Vektorgrafik (*vector graphics*)

¹Hexadezimalzahlen sind Zahlen im Sechzehnersystem (= Stellenwertsystem zur Basis 16). In diesem System werden die folgenden 16 Ziffern verwendet:

- 0, 1, 2, 3, 4, ..., 9 in der üblichen Bedeutung
- a steht für 10, b für 11, c für 12, d für 13, e für 14, f für 15.

Beispiel: Die Hexadezimalzahl `affe` entspricht der Dezimalzahl $10 \cdot 16^3 + 15 \cdot 16^2 + 15 \cdot 16 + 14 = 45054$

²Antworten:

- rot, grün, blau
- cyan (nahe an türkis), magenta (purpur), gelb
- ein relativ dunkles Blau; Man muss den Rot-Wert erhöhen, z.B. ist `#ae53df` ein Lila
- $256 \cdot 256 \cdot 256 = 2^8 \cdot 2^8 \cdot 2^8 = 2^{24} = 16'777'216$ Farben.

2.4. Rastergrafiken.

Dieser Abschnitt folgt stark einer Vorlage von Barbara Pampel.

2.4.1. Wenn man ein Bild, etwa die Kinderzeichnung in Abbildung 1 links, als Rastergraphik speichern will, legt man ein quadratisches Raster über das Bild, bestimmt für jedes Quadrat in diesem Raster eine geeignete Farbe (etwa die Durchschnittsfarbe oder die dominierende Farbe) und speichert diese.

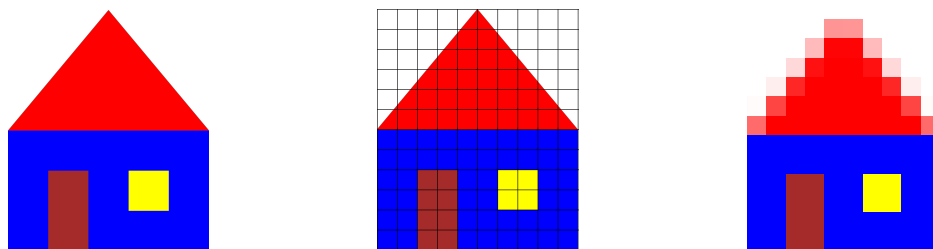


ABBILDUNG 1. Kinderzeichnung

Ein einfaches Datei-Format zum Speichern von Rastergrafiken ist das Datei-Format *Portable pixmap* mit der Datei-Endung *ppm*, das du in der folgenden Aufgabe kennenlernst.

Aufgabe 2.4.2. Speichere den unten folgenden Code (etwa mit Hilfe von VS Code oder eines geeigneten Text-Editor deiner Wahl, aber nicht mit Word!) in deinem Informatik-Verzeichnis unter dem Namen `haus.ppm`. Der Code ist auch online über [diesen Link](#) verfügbar und von dort vermutlich einfacher zu kopieren.

Öffne die Datei mit einem *pixelbasierten* Grafikprogramm, etwa mit [Adobe Photoshop](#) (kommerziell) oder [GIMP](#) (*Gnu Image Manipulation Program* (free and open source software)).

Das Bild ist vermutlich sehr klein, so dass du es stark vergrößern musst.

Schalte das Licht im Fenster auf Gelb und färbe die Tür braun oder manipulierte das Bild in einer anderen Weise, **indem du die ppm-Datei geeignet veränderst.**

Nicht erwünscht ist, dass du das Bild in Photoshop bearbeitest! Du sollst verstehen, welche Zahl in der ppm-Datei welche Bedeutung hat.

```
P3
# "P3" means this is a RGB color image in ASCII
5 6
255
# "255" is the maximum value for each color
# end of header, data below
255 255 255 255 255 255 255 0 0 255 255 255 255 255 255
255 255 255 255 0 0 255 0 0 255 0 0 255 255 255
255 0 0 255 0 0 255 0 0 255 0 0 255 0 0
0 0 255 0 0 255 0 0 255 0 0 255 0 0 255
0 0 255 255 255 255 0 0 255 255 255 255 0 0 255
0 0 255 255 255 255 0 0 255 0 0 255 0 0 255
```

2.4.3. Beispiele für Rastergrafik-Formate: *png* (Portable Network Graphics), *gif* (Graphics Interchange Format), *jpeg* (nach dem Gremium Joint Photographic Experts Group), *tif(f)* (Tagged Image File Format), *bmp* (Windows Bitmap), *ppm*.

2.5. Vektorgrafiken.

2.5.1. In der Vektorgrafik stellt man Bilder dar, indem man sie aus relativ einfachen Formen wie Linien, Kreisen, Polygonen, Ellipsen, Splines, Bezierkurven etc. zusammensetzt.

Ein gebräuchliches Format für Vektorgrafiken ist das *SVG-Format* (*scalable vector graphics* = skalierbare Vektor-Grafik).

Aufgabe 2.5.2. Speichere die unten angegebene *SVG-Datei* unter dem Namen `beispiel.svg` und zeige sie mit einem *vektorbasierten* Grafikprogramm an, etwa mit *Adobe Illustrator* oder [Inkscape](#). Der Code ist auch online über [diesen Link](#) verfügbar und von dort vermutlich einfacher zu kopieren.

Bringe das Dach in Ordnung (so dass es symmetrisch ist) und auch den blauen Bereich des Hauses (so dass es rechteckig ist). Färbe das Fenster gelb und ergänze eine braune Tür. **Auch hier ist die *svg-Datei* zu verstehen und zu verändern!** Bitte nicht im Grafikprogramm zeichnen.

Hinweise: Wo ist der Ursprung des Koordinatensystems? In welche Richtung zeigt welche Koordinatenachse? Mach eine Skizze auf Papier, bevor du loslegst!

```
<svg height="600" width="500">
  <polygon points="0,300, 400,0, 500,200"
    style="fill:red"/>
  <polygon points="0,300, 400,300, 500,600, 0,550"
    style="fill:blue"/>
  <polygon points="300,400, 400,400, 400,500, 300,500"
    style="fill:orange;stroke:cyan;stroke-width:20"/>
  <circle cx="80" cy="70" r="40" stroke="orange" stroke-width="5" fill="yellow"/>
  <text x="80" y="160" text-anchor="middle" fill="blue">
```

```
transform="rotate(-50.2 80,160)">SVG is great</text>
</svg>
```

2.5.3. Beispiele für Vektorgrafik-Formate: eps (Encapsulated Postscript), ps (Postscript), svg, wmf (Windows Metafile)

2.6. Vergleich: Rastergrafik versus Vektorgrafik.

2.6.1. Der Hauptunterschied liegt in der Skalierbarkeit: Vektorgrafiken lassen sich beliebig vergrößern und verkleinern und verlieren dabei nicht an Schärfe (im Gegensatz zu Rastergrafiken).

Vektorgrafiken werden hauptsächlich für Logos, Illustrationen und einfache Zeichnungen verwendet, Rastergrafiken für digitale Photographien und Reproduktionen (etwa Scans).

Meist ist der Speicherbedarf von Rastergrafiken deutlich höher als der von Vektorgrafiken. Die Erstellung von Vektorgrafiken ist meist komplexer als die von Rastergrafiken.

Vektorgrafiken lassen sich problemlos in Rastergrafiken umwandeln. Andersherum ist das meist kaum sinnvoll möglich (man denke etwa an ein Photo).

2.7. Zusatzinformationen.

2.7.1. Biologisch-physikalischer-technischer Hintergrund: Physikalisch gesehen ist das Farbspektrum kontinuierlich. Da die menschliche Farbwahrnehmung im Auge aber über nur drei Rezeptoren (**Zapfen**) geschieht, die grob gesagt auf gewisse rote, grüne und blaue Farbbereiche reagieren, kann man dem Menschen über das RGB-Farbmodell jede mögliche Farbe „vorgaukeln“, indem man Rot-, Grün- und Blauwert gerade so anpasst, wie sie bei der gegebenen Farbe vom Auge wahrgenommen werden würden. Dies macht die Produktion von Computerbildschirmen vergleichsweise billig (man stelle sich etwa vor, der Mensch hätte 20 oder noch mehr verschiedene Arten von Farbzapfen).

2.7.2. Neben dem RGB-Modell gibt es noch andere Farbmodelle, etwa CMYK (cyan, magenta, yellow, key=black, subtraktives Farbmodell, etwa zum Drucken) oder HSV (hue = Farbton, saturation = Sättigung, value = Dunkelstufe) oder HSL (hue, saturation, lightness = Helligkeit).

2.8. Bonusaufgaben.

Aufgabe 2.8.1. Schreibe eine ppm-Datei mit kleinen Abmessungen (Format etwa 4×2), so dass das entsprechende Bild alle $2^3 = 8$ RGB-Farben anzeigt, bei denen jeder der drei RGB-Werte entweder 0 oder 255 ist, vgl. [Farbbalkentestbild](#).

Wenn du willst, kannst du auch alle $3^3 = 27$ Farben mit RGB-Werten entweder 0 oder 127 oder 255 anzeigen.

Aufgabe 2.8.2. Zeichne das Haus vom Nikolaus (oder eine andere Zeichnung deiner Wahl) mit Hilfe einer SVG-Datei.

Beim Haus des Nikolaus ist der Befehl `polyline` nützlich, siehe [W3schools](#), [SVG tutorial](#), [SVG polyline](#). Von dort aus findest du auch leicht weitere Erklärungen zu SVG-Befehlen und diverse Beispiele.