

1. Du verstehst den Aufbau der Rasterbildformate PBM, PGM und PPM im ASCII-Format und kannst sie interpretieren. Im Folgenden ist ein *Leerraum* (*Whitespace*) eine Zeichenkette, die aus Leerzeichen, Tabulatoren oder Zeilenendzeichen besteht.
 - *Portable Bitmap (PBM)*: Magic Number (P1), Leerraum, Breite in Pixeln, Leerraum, Höhe in Pixeln, Leerraum, Bilddaten (0=Weiss, 1=Schwarz)
 - *Portable Graymap (PGM)*: Magic Number (P2), Leerraum, Breite in Pixeln, Leerraum, Höhe in Pixeln, Leerraum, maximaler Helligkeitswert, Leerraum, Bilddaten (ganze Zahlen von 0 bis und mit dem maximalen Helligkeitswert durch Leerraum getrennt; 0 beschreibt den dunkelsten Grauton)
 - *Portable Pixmap (PPM)*: Magic Number (P3), Leerraum, Breite in Pixeln, Leerraum, Höhe in Pixeln, Leerraum, maximaler Helligkeitswert, Leerraum, Bilddaten (Pro Farbe drei ganze Zahlen von 0 bis und mit dem maximalen Helligkeitswert, welche den Rot-, Grün- und Blauanteil der Farbe im RGB-Farbraum beschreiben. Dabei ist 0 jeweils der dunkelste Farbton).

Im Header (also vor den Bilddaten) sind Kommentare erlaubt, die durch das Symbol `#` am Zeilenanfang gekennzeichnet werden.

2. Du kannst RGB-Farben rudimentär interpretieren, wobei hier die einzelnen Werte für Rot, Grün und Blau durch ganze Zahlen zwischen 0 und 255 (inklusive) dargestellt werden. Dazu gehören die Farben.
 - Rot: (255, 0, 0)
 - Grün: (0, 255, 0)
 - Blau: (0, 0, 255)
 - Schwarz: (0, 0, 0)
 - Weiss: (255, 255, 255)
3. Du kannst einfache Bilder im PBM-, PGM- und PPM-Format interpretieren, d. h. aufgrund des Codes die Bilder zeichnen oder möglichst genau beschreiben.
4. Du kannst Python-Code schreiben, der einfache Bilder im PBM-, PGM- und PPM-Format erzeugt. Dazu musst du ...
 - (a) mit der `open(...)`-Funktion eine Datei mit dem gewünschten Namen zum Beschreiben öffnen, und den von dieser Funktion zurückgegebenen Dateideskriptor (Synonyme: *file descriptor* oder *file handle*) einer von dir gewählten Variable (z. B. `dd`, `fh` oder `fd`) zuweisen.
 - (b) den für die Bilder nötigen Code als Zeichenketten ("`...`" oder '`...`') mit der Methode `write(...)` in deinen Dateideskriptor schreiben. Beachte, dass anders als bei der Funktion `print()` Zeilenschaltungen manuell mit dem Steuerzeichen `\n` am Ende der Zeile angefügt werden müssen.
 - (c) die Datei (d. h. den Dateideskriptor) mit der `close()`-Methode wieder schließen, nachdem das letzte Zeichen in die Datei geschrieben wurde.