Für unser P5.js Projekt in Computational Thinking wollten wir eine festliche Karte im winterlichen Stil gestalten.

Codeerklärung:

* Zeile 1 – 4: auskommentierte RGB-Werte der Farben der Formen, für die wir uns entschieden haben.
* Zeile 7 – 9: ausgewählten RGB-Werte werden nach R, G und B Anteil in ein zugehöriges Array gespeichert, z.B.: R = redColour.
* Zeile 10 – 20: weitere Variablen für den späteren Programmablauf werden initialisiert.
* Zeile 22 – 26: *function setup()* wird aufgerufen, sie generiert den Hintergrund und mit *noloop()* wird eine ständige Wiederholung der *function draw()* verhindert.
* Zeile 29 – 31: mit der *function preload()* wird der Text als png-Datei aus dem Ordern „generative-kunst-team-13“ geladen.
* Zeile 33: *function draw()* wird vom Programm automatisch aufgerufen.
* Zeile 35 – 58: der obere Teil für die Formen auf der Karte wird generiert.
  + Zeile 35: Anzahl der zu erstellenden Formen wird anhand der Werte des Canvas ermittelt.
  + Zeile 36 – 44: *for-Schleife*, welche so oft die Befehle aufruft, bis sie die vorgebene Anzahl an zu erstellenden Formen erreicht hat.   
    Danach werden die Variablen für die Platzierung und Größe jeder einzelnen Form mithilfe von *random* in einem vorgegebenen Intervall ermittelt.
  + Zeile 47 – 51: eine Indexnummer wird per *random* aus dem Intervall unserer Anzahl an gewählten Farben zugewiesen. Dieser Index wird dann für jedes Farbarray eingesetzt und als den zugehörigen RGB-Wert eingesetzt. Zuletzt wird die Transparenz der Füllfarbe für die einzelne Form bestimmt.
  + Zeile 53 – 55: *if-Bedingung* welche die jeweilige y-Koordinate der Form abfragt, wenn sie einen bestimmten Wert überschreitet, wird die Transparenz der Form erhöht, damit die Schrift bei Überlappungen mit den Formen noch leserlich bleibt.
  + Zeile 57 – 58: *function form*() wird aufgerufen und der obere Teil ist fertig.
* Zeile 60 – 83: obere Teil von Zeile 35 – 58 wird wiederholt nur eben für den unteren Teil des Canvas. Hierbei unterscheiden sich hauptsächlich die Intervalle für die y-Koordinate sowie der Wert in der *if-Bedingung*, der Rest bleibt so weit gleich.
* Zeile 85 – 87: zuerst wird der *rectMode* auf *CENTER* gestellt, damit das Bild später mittig ist.  
  Danach wird die png-Datei mit unserem selbsterstellen Schriftzug dargestellt. Zuletzt ändern wir den *rectMode* wieder auf *CORNER*, damit die Positionen der Formen nicht beeinträchtigt werden.
* Zeile 92 – 106: *function form(x, y, e, c, r, g, b, transp)* wird mit bereits definierten Variablen aufgerufen.
  + Zeile 94 & 105: definieren den Anfang und das Ende unserer komplexen Form.
  + Zeile 96: der erste Ankerpunkt wird gesetzt, von dem die weiteren Ankerpunkte später weitergeführt werden.
  + Zeile 97 – 98: Farbe und Umrandung werden definiert.
  + Zeile 100 – 103: die weiteren Ankerpunkte werden gesetzt, zusätzlich wird der Grad der Kurve in der Form je nach erhaltenem Wert gleichmäßig verstärkt.
* Zeile 108 – 11: *function mousePressed()* wird aufgerufen, sobald das Programm einen Mausklick auf dem Canvas wahrnimmt. Ist dies der Fall, wird erneut der Canvas erzeugt und die *function draw()* aufgerufen. Wodurch eine neue Variante der Karte entsteht.