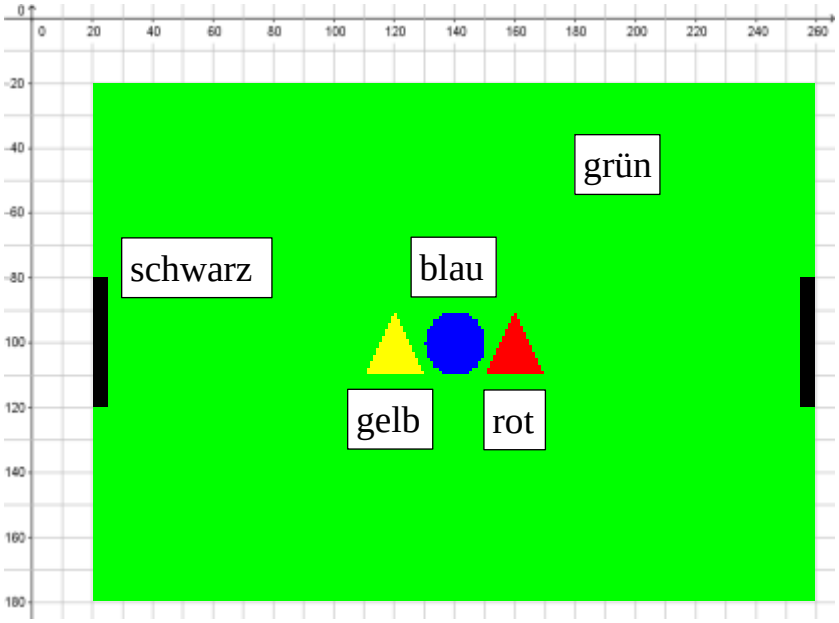
Programmieraufgabe Fussballspiel:

1. 💻 Verwende wieder das Projekt `2016_IOP_Vollstaendig` und speichert es unter einem neuen Namen, um ein Fußballspiel zu simulieren. Dazu benötigt man ein Spielfeld und Tore, dargestellt durch Rechtecke, und einen Fußball der Klasse `KREIS`. Definiere die nötigen Objekte in einer Klasse `FUSSBALLSPIEL`.

2. 💻 Erstelle den zugehörigen Konstruktor, der alle Objekte der entsprechenden Klassen anlegt.

3. 💻 Es ist einfacher, wenn die Objekte im Konstruktor von `FUSSBALLSPIEL` gleich alle nötigen Werte zugewiesen bekommen und es in den jeweils aufgerufenen Klassen entsprechende Konstruktormethoden gibt, also in `RECHTECK` eine, die auf `xPos, yPos, Breite, Höhe, Farbe` wartet. Verändere die Konstruktormethoden in den Klassen `KREIS` und `RECHTECK` und erzeuge Spielball, Spielfeld und die Tore. Mache dann alle Objekte sichtbar und teste Deine Methode. Hier kannst du die zu übergebenden Werte ablesen:



4. 💻 Wie man sieht, sollen die Spieler durch Dreiecke dargestellt sein. Erstelle nun in der Klasse `FUSSBALLSPIEL` zwei Spieler `trick` und `track` der Klasse `SPIELER`. Da es diese Klasse noch nicht gibt, erstelle sie jetzt. Ein Spieler hat als einziges Attribut ein Dreieck `d`. In `FUSSBALLSPIEL` legst Du jetzt `trick` und `track` mit den richtigen Werten an und übergibst die Werte für `xpos` und `ypos` der Spitze und `farbe` an `SPIELER`. Von dort werden sie an `DREIECK` weitergegeben. Hier musst Du den Konstruktor wieder anpassen. Höhe und Länge setzt Du dabei auf 20. Jetzt musst Du das zum Spieler

gehörige Dreieck noch sichtbar machen. Das machst Du natürlich im Konstruktor von `SPIELER`.

5. 💻 Wenn Du jetzt obiges Bild hast, dann geht es ans Wesentliche. Ein Spieler kann `schiessen()`, was bedeutet dass er den Ball eine bestimmte Strecke weit schiesst. Er bekommt also ein Objekt der Klasse `KREIS` und ein `int`-Wert übergeben, woraufhin das Objekt (also der Spielball) um den entsprechenden Wert nach links bzw. rechts verschoben wird. Erstelle die Methode, die du so aber noch nicht testen kannst.

6. 💻 Darum erstellst du in der Klasse `FUSSBALLSPIEL` eine Methode `spielen()`, die `track` den Ball um 40 Einheiten weit schiessen lässt.

7. 💻 Hat das geklappt? Dann verbessern wir die Methode jetzt. Aufgrund einer Zufallszahl-Fallunterscheidung (Zahl <> 0.5) entscheidest Du, ob `track` oder `trick` schiesst; natürlich in unterschiedliche Richtungen. Wieder geklappt? Dann entscheide mit einer weiteren Zufallszahl, wie weit (zwischen 10 und 50) einer der Spieler schiesst.

8. 💻 Verbessere die Methode `spielen()` weiter, indem du den Ball mit einer Wiederholung allmählich um den berechneten Wert verschiebst.

9. 💻 Außerdem müssen die beiden Spieler anschließend an die neue Position wechseln, sich also beide jeweils um den berechneten Wert `bewegen()`. Hier genügt es, wenn sie hinterher „springen“. Die in `FUSSBALLSPIEL` mit `spielen()` aufgerufene Methode muss auch in `SPIELER` programmiert werden, um dort eine Verschiebung des zum Spieler gehörigen Dreiecks um den übergebenen Wert zu bewirken.

10. 💻 Eine letzte Verbesserung der Methode `spielen()`: Überprüfe aufgrund der `xPosition` des Balls, ob ein Tor gefallen ist. Wenn ja, dann soll eine Meldung am Bildschirm erfolgen. Spieler und Ball werden anschließend in die Startpostition zurückversetzt. Ergänze dazu die Klasse `KREIS` um eine Methode `xposGeben()`, welche die `xPosition` zurückgibt und `KREIS` und `DREIECK` um die Methode `posSetzen()`, welche die Startwerte für `xPos` und `yPos` übergeben bekommt und die Objekte dort zeichnet. `posSetzen()` muss auch wieder in `SPIELER` ergänzt werden, um die Werte an die Objekte der Klasse `DREIECK` weiterzugeben.

11. 💻 Weitere Ideen:

    - Mitzählen des Spielstands und jeweilige Ausgabe
    - Zufallsfunktion, die einen Torwart den Schuss aufs Tor abwehren lässt