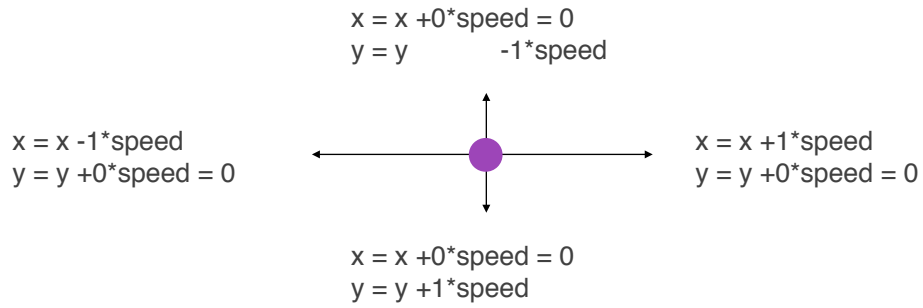
PACMAN 02

AUFGABE 02.1 - Gegner

Nun wollen wir Gegner mit einer kleinen AI erstellen. Kopiere dazu die Enemy Klasse:
https://github.com/fnues/BrueckenkursProgrammieren/blob/master/05Pacman/Enemy/Enemy.pde
Und füge sie deinem Spiel hinzu.
Die Klasse besitzt eine `move()` Methode. Diese soll für jeden `draw()` Durchlauf vor `Enemy.paint()` aufgerufen werden. Die move() Methode lässt den Gegner sich vertikal oder horizontal bewegen, solange er nicht auf ein Hindernis stösst. Das heisst wir ändern die x und y Koordinaten bei jedem move Aufruf. Dazu brauchen wir ein deltaX und deltaY, welche speichern in welche Richtung sich der Gegner bewegt. Sie können die Werte -1, 0, 1 annehmen (-1: Bewegt sich gegen den Nullpunkt, 0: bewegt sich nicht auf dieser Achse, 1: Bewegt sich entgegen dem Nullpunkt). Ist deltaX = -1, dann bewegt sich der Gegner nach links, da sich x wie folgt verändert: x = x+deltaX*speed, also x = x-1*speed. Wenn deltaX nicht gleich 0 ist, muss deltaY gleich 0 sein, damit eine vertikale Bewegung gewährleistet wird und umgekehrt.




```
class Enemy{
    float x;
    float y;
    int deltaX;
    int deltaY;
    float speed = 1;
    float enLength;
    float enWidth;
    PlayField obstaclesField;

    Enemy(float x, float y, float rectSize, PlayField field){
        this.x = x;
        this.y = y;
        this.obstaclesField = field;
        deltaX = 1;
        deltaY = 0;
        this.enLength = rectSize;
        this.enWidth = rectSize;
    }


// Enemy AI
void move(){
    // solange ein Hindernis des Playfield berührt wird, soll die Richtung
gewechselt werden
    while(field.isObstacleTouched(x + deltaX*enWidth/2, y + deltaY*enLength/2)){
        changeDirection();
    }
    // wenn nichts mehr berührt wird, soll x um deltaX*speed und y um
deltaY*Speed erhöht werden.
    x += deltaX*speed;
    y += deltaY*speed;
```

```
}


void changeDirection(){
    // weise deltaX einen zufälligen Wert von -1, 0 oder 1 zu
    deltaX = (int)random(0,3)-1;
    // falls deltaX == 0 ist, dann weise deltaY zufällig -1 oder 1 zu.
    if(deltaX == 0){
        int rand = (int)random(0,2);
        if(rand == 0) deltaY = -1;
        else deltaY = 1;
    // falls deltaX != 0 ist, soll deltaY = 0 sein.
    }else{
        deltaY = 0;
    }
}



    void paint(){
        fill(200, 0, 255);
        ellipse(x,y, enWidth, enLength);
    }


    boolean collides(float x, float y){
        // prüfe ob der Enemy berührt wird
       return (x > this.x && x < this.x+enWidth && y > this.y && y <
this.y+enLength);
    }
}
```

Erstelle im Main script eine ArrayList für Enemy.
Erstelle im Main script eine globale Variable `timeCount` welche bei jedem draw() Aufruf erhöht wird. Wenn der timeCount == 200 ist, soll der ArrayList ein neues Enemy Objekt hinzugefügt und der timeCount wieder auf 0 gesetzt werden. Füge nur solange enemies hinzu bis du 6 hast. Zeichne und bewege jeden enemy für jeden draw() Durchlauf und prüfe auch jedesmal ob einer mit Pacman kollidiert. Gib auf der Konsole: „du bist tod“ aus, wenn er kollidiert.

Die Enemies sollen jeweils in der linken oberen Ecke starten:

```
new Enemy(2*rectSize+rectSize/2, 2*rectSize+rectSize/2, rectSize, field);
```

AUFGABE 02.2 - Score

Nun wollen wir die Elemente erstellen, die Pacman einsammeln kann und somit seinen score erhöhen. Ich nenne diese Elemente Nuggets. Ein Nugget soll als kleiner blauer Kreis gezeichnet werden und die Methode `boolean getsEaten(float x, float y)` besitzen. Diese funktioniert ähnlich wie die `isTouched()` Methode, sobald Pacman ein Nugget berührt wird es „gegessen“. Wenn ein Nugget gegessen wurde, soll es nicht mehr gezeichnet also nicht mehr sichtbar sein.
Ergänze dazu die folgende Klasse:

```
class Nugget{
    // Attribute für Position und Grösse (setzte Grösse = 5)
    // Attribut für visible Zustand. Also Attribut das speichern kann ob ein
Nugget visible ist oder nicht

    Nugget(float x, float y){
        this.x = x;
```

```
        this.y = y;
    }


    void paint(){
        // zeichne einen blauen Kreis, falls das Nugget visible ist
    }


    boolean getsEaten(float x, float y){
        // teste ob das Nugget gegessen wird / ob der Pacman das Nugget berührt
        // vergesse nicht, dass es nicht mehr gegessen werden kann, wenn es
nicht visible ist
    }


    void setVisible(boolean visible){
        // eine Methode um das visible Attribut zu setzen.
    }

}
```

Erstelle eine globale `score` Variable im Main script, welche erhöht wird, wenn ein Nugget gefressen wird. Mach deinen Score sichtbar indem du ihn mit `text()` immer anzeigst. Erstelle nun ein Nugget Objekt und teste ob alles funktioniert.

Wir können dem Playfield nun auch einen Array von Nuggets hinzufügen und die Klasse mit einer Methode ergänzen, die prüft ob eines der Nuggets gegessen wurde und dieses dann invisible setzt. Kopiere die neue Playfield Klasse und ersetze damit deine alte:
https://github.com/fnues/BrueckenkursProgrammieren/blob/master/05Pacman/PlayField/PlayField.pde

Hier ein Ausschnitt der Klasse mit dem für die Nuggets relevanten Code:

```
class PlayField {
    // hier wird nur gezeigt was wir für Nuggets brauchen, nicht was wir schon
vorher hatten
    float rectSize;
    int nrOfElementsWidht = 9;
    int nrOfElementsHeight = 7;
    float startFieldX;
    float startFieldY;

// initialisiere Nugget Array so dass es eine Array Element für jedes Grid
Element des Spielfelds gibt
    Nugget[] nuggets = new Nugget[nrOfElementsWidht*nrOfElementsHeight];

    PlayField(float rectSize) {
        this.rectSize = rectSize;
        this.startFieldX = rectSize;
        this.startFieldY = rectSize;

        initNuggets();
    }


    // in jedem Gridelement wird ein Nugget in die Mitte Plaziert
    void initNuggets(){
        int nuggetCount = 0;
        for ( int x = 0; x < nrOfElementsWidht; x++ ) {
            for ( int y = 0; y < nrOfElementsHeight; y++ ) {
                nuggets[nuggetCount] = new
Nugget(startFieldX+x*rectSize+rectSize/2, startFieldY+y*rectSize+rectSize/2);
                nuggetCount++;
            }
        }
```

```
    }


    void paint(){
        // zeichne zuerst die Nuggets und dann die Hindernisse
      for(int i = 0; i < nuggets.length; i++){
            nuggets[i].paint();
      }
    }


    boolean isOneNuggetEaten(float x, float y){
        // prüfe ob eines der Nuggets gegessen wird, falls ja, setzte dieses
invisible
      for (int i = 0; i < nuggets.length; i++) {
            boolean getsEaten = nuggets[i].getsEaten(x, y);
            if(getsEaten){
                  nuggets[i].setVisible(false);
                  return true;
            }
      }
return false;

    }

}
```

Rufe die isOneNuggetEaten() Methode des Playfields in der draw Methode des Main Scripts auf. So dass bei jedem Durchlauf geprüft wird ob ein Nugget gegessen wird. Wenn ja wird es invisible gesetzt und der Score hochgezählt.

AUFGABE 02.3 - Spiel abrunden: Game start, Game over

Füge im Main script eine gloable Variable hinzu welche prüft ob Pacman sich schon bewegt hat und eine die prüfen kann ob das Spiel fertig ist, also ein Gegner berührt wurde.
Nutze die erste Variable dazu, dass du erst Gegner generierst, sobald sich Pacman zu bewegen beginnt.
Und nutze die zweite dazu, dass Spiel abzubrechen, wenn ein Gegner berührt wurde. Zeichne dazu nichts mehr vom Spiel sondern zeige nur einen Text „Game Over“ und vielleicht noch welchen Score erreicht wurde.

```
boolean isGameOver = false;
boolean pacmanStarted = false;

Start:
```

Game Over: