Aufgaben zur Abgabe als Prüfungsleistung für Prog I

- Allgemeine Regeln
  - Es gelten dieselben Regeln wie für Aufgaben 1 und 2, nur ein anderer Abgabetermin
  - Aber: Jetzt soll alles objektorientiert programmiert werden, mit Arrays, aber ohne Collections

3. Aufgabe: Snake

- Snake ist ein klassisches Spiel aus dem 70er Jahren, in dem der Spieler eine Schlange steuert, um einen Apfel bzw. Nahrung auf dem Spielfeld zu erreichen
  - Die Steuerung der Schlange erfolgt über die 4 Pfeiltasten
  - Falls die Schlange über den Spielfeldrand hinauskommt oder den eigenen Körper berührt, ist das Spiel beendet
  - Da wir zum jetzigen Zeitpunkt noch keine GUI-Entwicklung besprochen haben, wird die grafische Ausgabe mittels Hilfsklassen unterstützt, die auch verwendet werden müssen
  - Es gibt zwei mögliche Versionen dieser Aufgabe:
    - Version 1 erreicht maximal 12 Punkte (7 Punkte Funktionalität, 5 Punkte Qualität)
    - Version 2 erreicht maximal 20 Punkte (12 Punkte Funktionalität, 8 Punkte Qualität)

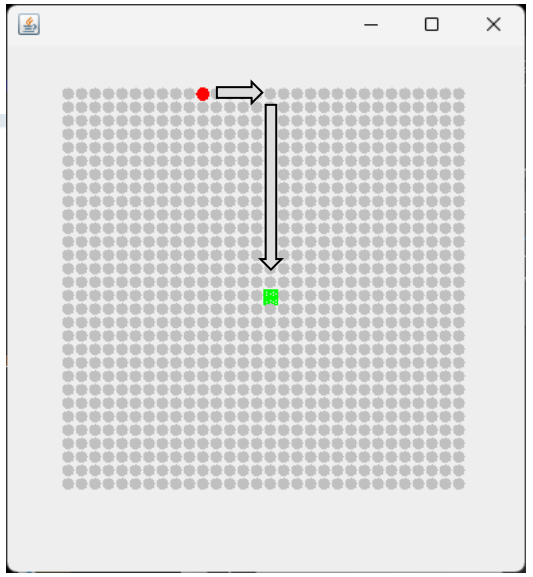
https://commons.wikimedia.org/wiki/Category:Snake_(video_game)?uselang=de#/media/File:Suomen_Pelimuseo_-_matopeli.jpg

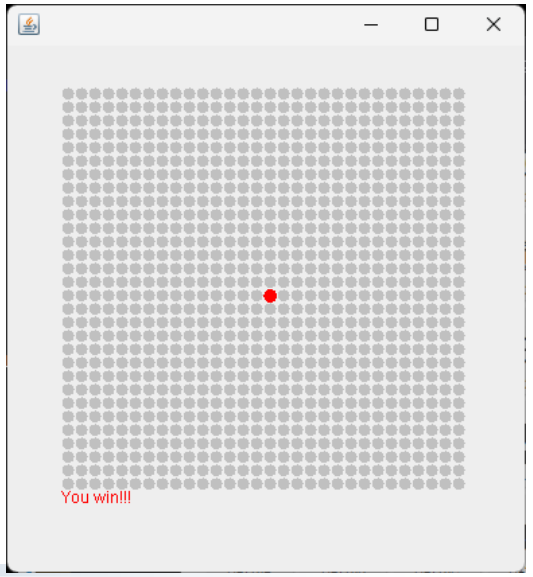
Vorgaben

- Die Implementierung startet mit den gegebenen Klassen `Main.java` und `PlayingField.java`
  - Klasse `Main.java`
    - Startet das Spiel mittels Timer-Steuerung (z.B. Aktualisierung jede 500ms)
  - Klasse `PlayingField.java`
    - Verwaltet u.a. die Daten zum Spielfeld (= zweidim. Feld), zur Schlange (Klasse `Snake.java`) und zur Nahrung (`Food.java`) mittels Komposition
    - Im Spielfeld laufen die x-Koordinaten 0,1,.. nach rechts, y-Koordinaten 0,1,… nach unten
    - Die Abfrage der letzten gedrückten Pfeiltaste gibt der Schlange eine Richtung
  - `Snake` und `Food` besitzen u.a. die Gemeinsamkeit, dass beide eine Position auf dem Spielfeld besitzen, dies soll mittels Vererbung über eine Klasse `GameObject` abgebildet werden
    - Die Position soll den bereits in den Java-Bibliotheken vorhandenen Datentyp `Point` verwenden, d.h. `import java.awt.Point;` bietet einen Zugriff auf x- und y-Koordinate, d.h. `column` und `row` im Spielfeld
    - Die aktuelle Position der Schlange wird über den Kopf der Schlange definiert, d.h. wenn der Kopf der Schlange das Food-Objekt erreicht, ist ein (Zwischen-)Ziel erreicht

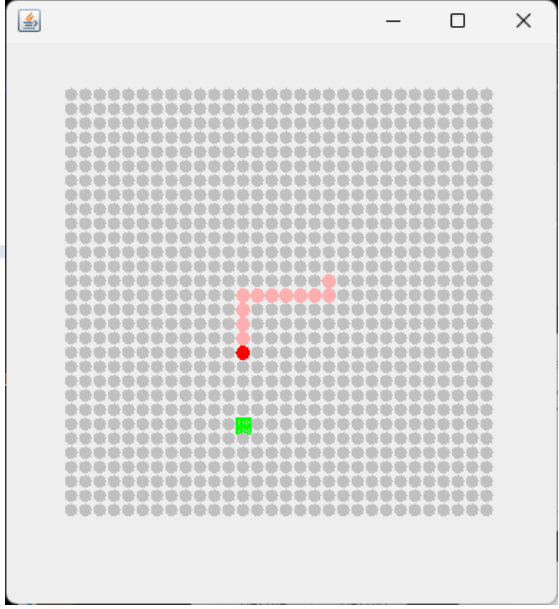
Erste Version der Anwendung

- In der ersten Version soll die Schlange noch keinen Körper besitzen, sondern nur den Kopf auf einer Startposition des Spielfelds (= feste Position 0/0)
  - In jedem Schritt bewegt sich die Schlange gemäß Pfeiltasten am Anfang automatisch nach rechts (bereits vorhandenes enum `Direction.RIGHT` etc. verwenden)
  - Wenn der Spieler das Food-Objekt erreicht hat (= feste Position Mitte Spielfeld), soll das Spiel beendet werden (Anzeige Meldung, siehe Abbildung)
  - Wenn die Schlange aus dem Spielfeld hinausläuft (z.B. x-Position < 0), wird das Spiel ebenfalls mit der Anzeige einer Meldung beendet
  - Farben und Layout wie im Screenshot, Symbole nach eigenem Geschmack, Hinweise links unten
- Mit dieser ersten Version können max. 12 Punkte erzielt werden





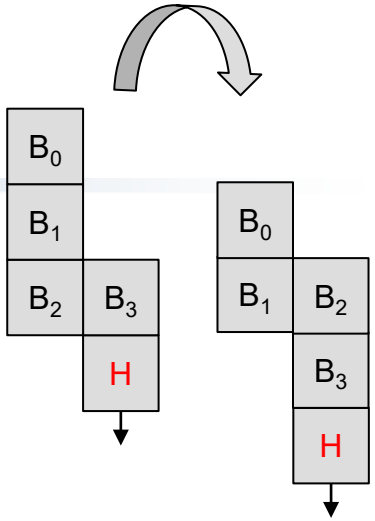
Zweite Version der Anwendung



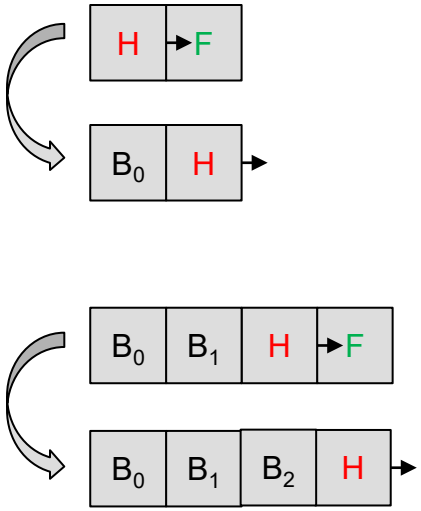
- Zum Erreichen der restlichen Punkte müssen folgende Erweiterungen umgesetzt werden
  - Die Schlange startet nur mit dem Kopf und erhält bei jedem Erreichen des Food-Objekts ein neues Körper-Segment, d.h. der Körper wird innerhalb der Klasse `Snake` in einem Feld von Points verwaltet, das im Laufe der Zeit immer mehr Elemente hat
    - Nach dem Verzehr des Food-Objekts wird automatisch ein neues Food-Objekt erzeugt, das an einer Zufallsposition außerhalb der Schlange positioniert wird

      ```
      int randomNumber = randomGenerator.nextInt(SIZE);
      ```
  - Die Schlange zieht während des Spiels ihren aktuellen (pinken) Körper hinter sich her
  - Die Schlange darf sich nicht selbst berühren, d.h. wenn der Kopf ein Element des Körpers berührt, ist das Spiel verloren
  - In der Fußzeile soll eine Erfolgsmeldung bei Erreichen des Foods für einen Zeitschritt angezeigt werden; weitere Meldungen bei Fehlern

Implementierung der zweiten Version



- Prinzip der Bewegung
  - H ist der Head der Schlange, d.h. die aktuelle Position
  - B_n sind die Body-Segmente der Schlange
  - Pfeil ist die aktuelle Bewegungsrichtung
    - Head bewegt sich einen Schritt gemäß Bewegungsrichtung
    - Die Body-Segmente werden verschoben, das vorderste Body-Segment nimmt die alte Head-Position ein
- Prinzip des Hinzufügen eines Body-Segments
  - F ist das Food
  - Wenn noch kein Body existiert, wird bei Erreichen des Foods ein erstes Body-Segment erzeugt
  - Wenn bereits Body-Segmente existieren, dann wird ein neues Segment ergänzt, d.h. das Body-Feld wird um ein Element länger



Umsetzung

- Spielzustand: Der aktuelle Zustand (verloren, gewonnen usw.) muss in einem `enum` verwaltet werden
- Beispiel-Algorithmus zur Erzeugung eines neuen Segments (darf auch anders implementiert werden)
  - Merke die vorige Position des Heads in einem Attribut der Snake-Klasse (wird jedesmal aktualisiert, wenn sich die Schlange bewegt)
  - In der Methode zum Hinzufügen eines Body-Segments:
    - Falls der body noch `null` ist, erzeuge ein neues Punkt-Objekt mit der gemerkten Koordinate. Dann wird ein body-Feld erzeugt und mit diesem Punkt-Objekt befüllt
    - Sonst erzeuge ein neues body-Feld mit einer um 1 größeren Länge als bisher;
      Kopiere die Punkte des alten body in den neuen body
      Füge den neuen Punkt aus der gemerkten Koordinate hinzu
- Vorlagen
  - Es gibt als Vorlage ein Paket zum Zeichnen (`package drawing`), dessen Klassen nur nach Rücksprache verändert werden sollen
  - Auch das Hauppprogramm `Main.java` nur nach Rücksprache verändern
  - `PlayingField.java` darf beliebig verändert werden