**Snake – Konzeptentwurf**

Spielfeld:

* Symmetrische Matrix bestehend aus der Menge {0,1,2}
  + 0: Randfeld
  + 1 Durch Snake besetztes Spielfeld
  + 2 : Freies Spielfeld
  + 3: Durch Snack besetztes Spielfeld
* Spielfeldrand:
  + Der Rand des Spielfeldes wird durch 1en gekennzeichnet. Entsprechend wird bei der initialisierung des Spielfeldes die Spalten und Zeilenanzahl jeweils um 2 erhöht, um die absolut gewünschte Spielfläche beizubehalten.

Gamelogic:

* Start des Spiels:
  + Spielfeld wird mit 0 initialisiert
  + An eine zufällige Position wird eine 0 zu einer 1 geändert – Startfeld von Snake
  + An einer zweiten zufälligen Position wird eine 0 zu einer 2 geändert.
    - Bedingung: Das Spielfeld muss 0 sein und darf nicht durch Snake besetzt sein
* Spielverlauf:
  + Wenn Snake sich auf ein Spielfeld befindet, dass == 2 ist, dann wird die 2 zu einer 1, und Snake wird um ein Spielfeld länger
  + Bevor Snake sich nach vorne bewegt, wird überprüft ob das Feld in der gewünschten Richtung == 1 oder == 0 ist. Wenn das Spielfeld == 1 ist, dann wird das Spiel beendet. Wenn das Spielfeld == 0 oder 2 ist, dann wird das Spiel fortgesetzt und die entsprechende Bewegung durchgeführt.
* Bewegung von Snake:
  + Der „Kopf“ von Snake bestimmt die Richtung in die sich Snake bewegt. Dabei gibt es drei Bewegungsrichtungen:
    - Nach Vorne: Wenn der Spieler keine Tasteneingabe tätigt bewegt sicht Snake nach vorne
    - Nach Links: Wenn der Spieler die linke Pfeiltaste drückt bewegt sich Snake nach links
    - Nach Rechts: Wenn der Spieler die rechte Pfeiltaste drückt bewegt sich Snake nach rechts
  + Die Position des Schwanzes von Snake wird durch die Position des vorhergehenden Schwanzstückes bestimmt.
    - Der Kopf wird auf die neue Position gesetzt in der gewünschten Bewegungsrichtung
    - Alle Teile die nicht der Kopf sind werden als Schwanzteil identifiziert
    - Die Position eines Schwanzteils wird durch die vorherige Position des vorhergehenden Körperteils von Snake bestimmt.
    - Nachdem der Kopf also bewegt wurde wird für alle wobei l die Länge von Snake ist die Position neu gesetzt
* Spielende:

Das Spielende tritt ein, wenn sich Snake im im folgenden Spielschritt auf ein Spielfeld bewegen würde, welches den Wert 1 hat (also der Rand oder Snake selbst)

Snake:

* Snake hat eine Exemplarvariable Länge, welche die gesamte Länge von Snake festhält
* Snake besitzt eine Exemplarvariable vom Typ LinkedList, die Objekte vom Typ Tail enthalten

Tail:

* Tail beeinhaltet vier Exemplarvariablen
  + currentX
  + currentY
  + previousX
  + previousY
* Bausteine von Snake

UI:

**Optimierung der Performance**

* Bodypart Klasse Entfernen und alternative Implementation der prev und curr position values verwenden
  + Arbeitsweise von adjustPosValues:
    - Iteriert über jeden Körperteil von Snake und setzt den Wert von prev auf curr und und curr auf prev des vorhergehenden Körperteils
  + Optimierungspotenzial:
    - Nutzen zweier Listen – eine mit x und eine mit den y werten.
    - Problem: Es werden ebenfalls Integer Objekte erzeugt