Planung des Vorgehens (Zeitplan, Arbeitsteilung, Milestones)

*\_* Konzeption des Spieles

*\_* Beschreibung besonderer Funktionen oder Libraries

*\_* Geeignete Testfälle für das Spiel, welche extern gespeichert werden können (pdf, doc, txt)

[CG] Aufgabenblatt S

|  |  |
| --- | --- |
| **Konzeption** | Ähnlich zu Beispielprogramm, ein Läufer in einem Tubus der Löcher im Tubus ausweichen muss. Wenn er herunterfällt verliert er ein Leben. |

|  |  |  |
| --- | --- | --- |
| **Milestone I – erste Woche** | | |
| **Klasse** | **Funktion** | **Bearbeiter** |
| **Tubus** | Erstellen eines Tubus mit x Ecken und y Segmenten pro Seite, muss eine Funktion enthalten um den Tubus in Richtung Kamera zu bewegen (moveZ) | Sebastian Kriege |
| **Exampleprimitives** | Hauptklasse zum Anzeigen von Objekten, Erstellen einer Instanz des Tubus, Aufruf der moveZ Funktion im Loop |  |

|  |  |  |
| --- | --- | --- |
| **Milestone II – zweite Woche** | | |
| **Klasse** | **Funktion** | **Bearbeiter** |
| **Renderable** | Erstellen einer Basisklasse für alle Objekte, die gerendert werden müssen. Diese Klasse übernimmt viele Funktionen von Exampleprimitives. Somit kann jedes Objekt einen anderen Shader verwenden und eine eigene Textur. |  |
| **Insect (extends Renderable)** | Unser Läufer |  |
| **Level** | Eine Klasse um Level zu erstellen. Da der Tubus aus Segmenten besteht ist es sinnvoll als Datenstruktur eine Matrix zu verwenden, wobei der erste Index das Segment angibt und der zweite Index die Position im Segment. |  |
| **GUI (extends Renderable)** | Eine Klasse zum Erstellen von 2D-Overlays. |  |
| **Tubus (extends Renderable)** | Abändern vom Tubus, so dass er auch über Renderable verwaltet wird und die Level angezeigt werden |  |

|  |  |  |
| --- | --- | --- |
| **Milestone III – dritte Woche** | | |
| Klasse | Funktion | Bearbeiter |
|  |  |  |