4.3 Spiel: Projekt Labyrinth

Aufgabe 4.3.1:

Erzeugen Sie selbst mittels eines Text-Editors weitere Konfigurationsdateien für das Labyrinth-Spiel und testen Sie sie im Spiel.

Aufgabe 4.3.2:

Erweitern Sie die Labyrinth-Datenstruktur und die Dateistruktur dahingehend, dass in der Konfigurationsdatei eine erlaubte Maximalzahl von Zügen festgelegt wird, beispielsweise über den Kenner „m“. Zählen Sie während des Spiels die bereits erfolgten Züge und stoppen Sie das Spiel, wenn die erlaubte Maximalzahl überschritten wurde.

Aufgabe 4.3.3:

Erweitern Sie das Labyrinth-Spiel, indem Sie die Sichtbarkeit des Spielfelds auf die Nachbarfelder der Spielfigur einschränken (Nebeleffekt). Alle Nicht-Nachbarfelder werden ohne Struktur gleichmäßig grau gezeichnet. Wird die Figur bewegt, wandert natürlich auch die Sichtbarkeit zur neuen Position.

Aufgabe 4.3.4:

Analysieren Sie die Methode *lese_spiel*, die die Spieldaten aus einer Textdatei einliest. Schreiben Sie eine ähnliche Funktion, die die Textzeilen einer Datei mittels *sscanf* in ihre Bestandteile zerlegt.

Aufgabe 4.3.5:

In dieser Aufgabe soll ein Männchen einen rechteckigen Rundgang ausführen. Sie benötigen dafür ähnliche Eigenschaften und Methoden wie für die Klasse *Labyrinth*, diesmal als Klasse *Rundgang* in einem separaten m-File, also die Properties *axes*, *p*, *feld*, *imag*, *fig* und den Konstruktor *Rundgang* und die Methoden *data_init*, *data_clear*, *draw_field*, *leeres_feld* und *zeichne_figur*. Die Daten des Grafikfelds liegen diesmal nicht in einem Konfigurations-File, sondern sind fest vorgegeben, z.B. `zeilen = 10;` und `spalten =12;`.

Erstellen Sie dafür im App Designer eine neue App mit einem Grafikfeld, analog unserem Labyrinth-Spiel. In der *startupFcn* wird ein Objekt vom Typ Rundgang erzeugt. Dann wird ein Timer gestartet, der als Callback-Funktion die Methode *laufe* von *Rundgang* aufruft, die die neue Position des Männchens berechnet und das Männchen zeichnet.