# 1 Funktionen

## 1.1 Bin ich drin?

Prüfe ob ein Wert im Intervall liegt. Benötigte Operatoren für Zahlen $<$, $>$ und $=$. Außerdem and und or für die Verknüpfung von boolschen bzw. Wahrheitswerten und if für die Auswertung.

## 1.2 Berechne die Summe aller Zahlen von 1 bis n

Die Umsetzung sollte mit einer Schleife passieren. Welche Möglichkeiten fallen dir noch ein?

## 1.3 !murehtrhekreV: Gibt den Text rückwärts aus!

Als Übergabewert gibt es einen Text als string. Von diesem kann man mit Hilfe der funktion Length() die Länge bestimmen. Außerdem kann man mit dem Operator $[i]$ auf eine Buchstaben im Text an der Stelle i zugreifen. Beachte, dass strings bei Index 1 beginnen nicht bei 0 wie Arrays. Weiterhin wird der Operator + benötigt um Texte zusammen zu setzen.

## 1.4 Zaehle alle Vokale!

Prüfe jeden Buchstaben im Text ob es ein Vokal ist und gib deren Anzahl zurück. Beachte Groß und Kleinbuchstaben.

## 1.5 Buchstabiere das Wort!

Wandle das übergebene Wort in eine durch Leerzeichen getrennte Aussprache seiner Buchstaben um. Aus dem Text „abcdjky“ wird zum Beispiel „a be ce jot ka ypsilon“.

## 1.6 Sage die Ziffern!

Wandle die übergebene Zahl in den entsprechenden Text bei der Aufzählung der Ziffern getrennt durch Leerzeichen um. Hilfreiche Operatoren sind div und mod sie geben jeweils den ganzzahligen Teil einer Division (div) und den Rest (mod) zurück. Aus 1234 wird z.B. „eins zwei drei vier“. Welches ist die beste Reihenfolge um die Zahl auseinander zu nehmen?

## 1.7 Berechne die Summe!

Eine Folge von Zahlen wird als array übergeben. Addiere alle Zahlen im array zusammen und gib das Ergebnis zurück. Die Länge des Array kann man mit Length bestimmen. Der erste Eintrag des Arrays beginnt bei 0.

## 1.8 Welche Zahl in der Folge fehlt?

Als Übergabewert gibt es eine Folge von Zahlen in Form eines Arrays. Die Zahlen sind sortiert und sollten sich immer um 1 unterscheiden. Gib die fehlende Zahl zurück falls in der Folge ein Sprung größer als 1 Auftritt. Gibt es Keinen, gib eine 0 zurück.

## 1.9 Begrüße mich!

Als Übergabewerte gibt es eine Tageszeit in Form eines Aufzählungstyps Tageszeit und einen Namen. Je nach Tageszeit soll eine andere Begrüßung als Rückgabewert entstehen.

| | |
|---|---|
| morgens | Guten Morgen |
| mittags | Mahlzeit |
| abends | Guten Abend |
| nachts | Gute Nacht |

Z.B. wird aus morgens und Bernd „Guten Morgen, Bernd!“. Benutze eine case Anweisung zur Unterscheidung der Tageszeit.

## 1.10 Berechne die Hypotenuse!

Als Übergabewerte gibt es die Länge der beiden Seite am rechten Winkel eines Dreieck. Welche Funktionen benötigt man für Wurzeln und Quadrate von Zahlen?

## 1.11 Sag's mir später.

Gib den Text zurück der beim vorherigen Aufruf der Funktion übergeben wurde. Gib beim ersten Aufruf einen leeren string zurück. Was ist die Schwierigkeit hieran?

## 1.12 Sortiere die Liste

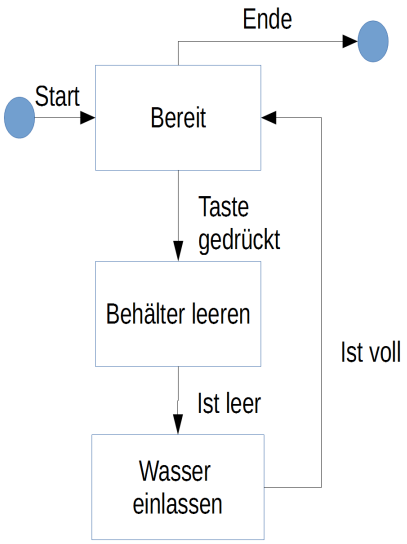
Sortiere die Zahlen im übergebenen Array von klein nach groß. Ein einfacher Sortieralgorithmus ist Bubblesort informiere dich dazu und setze den Algorithmus um. Dieser Algorithmus muss mehrfach über die zu sortierenden Daten laufen. Was ist nach dem ersten Durchlauf passiert?

# 2 Zustandsdiagramme und Roboter

## 2.1 Zustandsdiagramm

### 2.1.1 Beispiel Spülung

Mit einem Zustandsdiagramm kann man die Zustände eines Objektes oder Systems abbilden. Mit den Rechtecken werden die Zustände abgebildet in denen sich ein System befindet. An den Pfeile stehen die Ereignisse die auftreten und einen Wechsel des Zustandes des Systems verursachen. Außerdem gibt es Punkte die den Start und die Beendigung markieren.

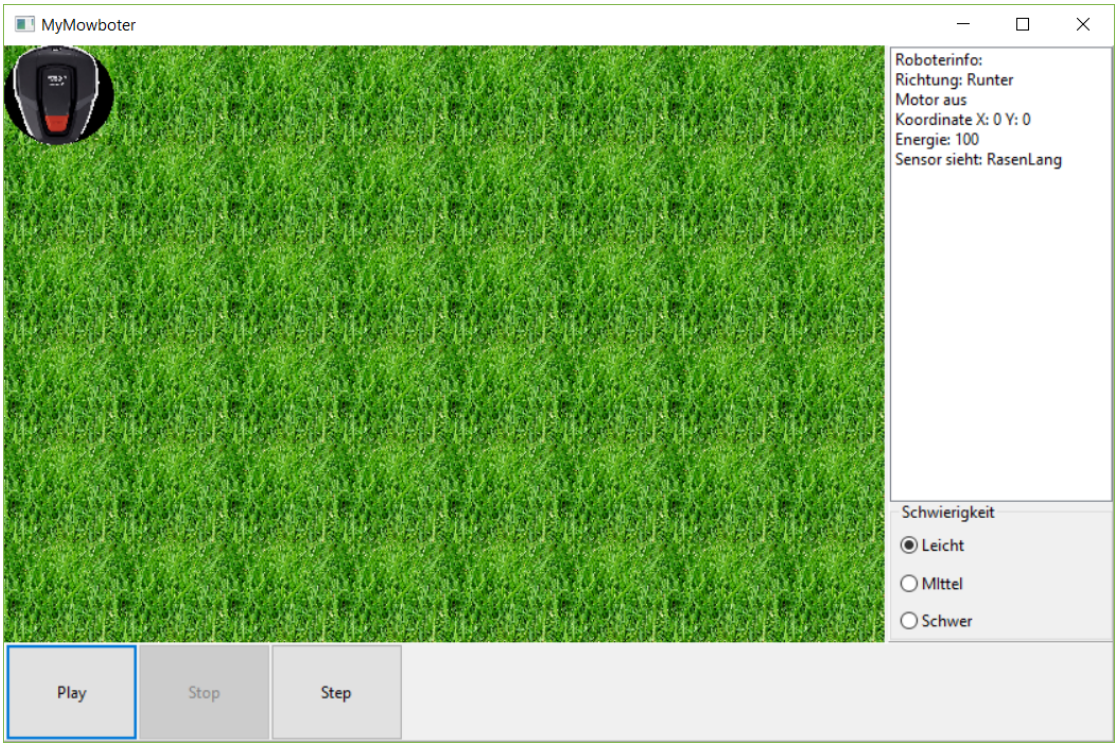



Im Beispielbild ist das Diagramm einer Klospülung abgebildet. Zu Begin ist das System im Zustand bereit. Wird dann die Taste betätigt wird damit begonnen den Wasserbehälter zu entleeren. Wenn der Behälter leer ist, wird mit dem Füllen des Behälters begonnen. Wenn dieser komplett befüllt ist, befindet sich das System wieder im Bereit Zustand und kann eine weitere Spülung ausführen.

### 2.1.2 Übungsaufgabe Kaffeeautomat

Entwerfe das Zustandsdiagramm für einen Kaffeeautomaten. Dieser befindet sich zunächst im Bereit-Zustand. Danach wird mit dem Startknopf die Zubereitung gestartet. Dazu werden zuerst die Bohnen gemahlen. Ist das abgeschlossen wird das Pulver gepresst. Nach Beendingung dessen wird damit begonnen das Wasser durch das Pulver zu pressen. Das passiert solange bis die benötigte Wassermenge erreicht ist oder der Stopknopf gedrückt wird. Danach wird das benutzte Pulver in den Abfallbehälter befördert. Ist das abgeschlossen befindet sich der Automat wieder im Bereit-Zustand.

## 2.2 Mähroboter




Implementiere die künstliche Intelligenz eines Mähroboters der den Rasen auf dem Spielfeld mähen soll. Im Verzeichnis Rasenmaeher ist das Projekt Rasenmaeher. Wenn man das Projekt startet sieht man zunächst die Oberfläche 2.2 mit dem Rasen und dem Roboter. Außerdem hat man rechts einige Informationen zum Roboter. Unten sind Buttons um den Spielverlauf zu steuern. Man kann den Roboter automatisch laufen lassen oder einzelne Schritte ausführen.

Der Roboter darf nicht außerhalb des Bereichs fahren. Außerdem darf auf einem bereits gemähten Feld nicht erneut gemäht werden, damit der Rasen nicht kaputt geht. Außerdem darf das Blumenbeet auch nicht befahren werden.

Rechts unten kann man die Schwieriegkeit einstellen. Auf leicht startet der Roboter immer oben links. Auf Mittel startet der Roboter an einer zufälligen Position und mit einer zufälligen Richtung. Auf schwer gibt es zusätzlich ein Blumenbeet auf das man nicht fahren darf.

In der Unit uAI soll das Verhalten des Roboters beeinflusst werden. Vor jedem Schritt wird die Funktion on NextStep aufgerufen und man bekommt den Roboter als Übergabeparameter um seine Richtung zu beeinflussen (property Richtung) und den Motor zu starten (property MotorAn). Außerdem kann man mit der funktion BenutzeSensor den Typ des Feldes herausfinden, das sich vor dem Roboter befindet.

Das Spiel ist dann gewonnen, wenn alle Felder gemäht wurden.

Wenn das Spiel gewonnen wurde, entwirf das Zustandsdiagramm für das Verhalten des Roboters.

# 3 Übungen Oberfläche

Für die nachfolgenden Aufgaben lege im Lazarus unter Datei Neu Anwendung eine neue Anwendung an und speichere sie in einem Ordner. Alle Oberflächenelemente einer Aufgabe sollen zusammen in einer Groupbox sein die den Namen der Aufgabe bekommt (zu finden unter Standard). Die Aktionen werden meistens über das OnClick Ereignis der entsprechenden Buttons ausgeführt. Wichtig ist es die Namen der Oberflächenelemente zu kennen nachdem man sie hinzugefügt hat, damit man später Aktionen mit ihnen durchführen kann.

## 3.1 Blinky

Ziel ist es ein Shape blinken zu lassen. Hierfür benötigen wir einen Shape (unter Additional), einen Button (Standard) und einen Timer (System). Ein Timer wiederholt die Prozedur die als OnTimer zugeordnet sind mit dem im Intervall eingestellten Abstand. Das kann man mit Doppelklick anlegen oder unter Ereignisse OnTimer. Mit der Eigenschaft Enabled kann man den Timer starten oder stoppen. Damit der Shape blinkt soll ihm abwechselnd die Farbe Blau und Rot zugewiesen werden. Das geschieht über die Eigenschaft Brush und Color. Die Farben sind Konstanten Ihren Namen kann man im Objektinspektor herausfinden. Dazu muss zu beginn die Farbe des Shapes auf Blau sein (im Objektinspektor einstellen). Mit dem zugehörigen Button soll der Timer gestartet oder gestoppt (Enabled auf true oder false) werden können um das Blinken an- und auszuschalten. Dieser soll die Caption Start oder Stop bekommen je nachdem ob der Timer an oder aus ist.

## 3.2 Text

Für diese Aufgabe wollen wir Text wollen wir Text in ein Memo laden und ihn wieder abspeichern. Sobald etwas im Memo steht kann man es beliebig editieren.

Dazu benötigen wir zuerst ein Memo, ein paar Buttons und Dialoge. Um eine Datei zu laden benötigen wir zunächst einen OpenDialog (unter Dialoge) und einen Button „Laden“ um ihn aufzurufen. Für den Dialog stellen wir einen Filter für Text Dateien ein der nur *.txt filtert. Einen Dialog kann man mit Execute aufrufen. Den Ausgewählten Dateinamen bekommt man mit der Eigenschaft FileName. Um den Inhalt des Memos zu bearbeiten greifen wir auf Memo.Lines zu was vom Typ Stringliste ist. Mit LoadFromFile kann man diese direkt aus einer Datei laden.

Zum Speichern benötigen wir einen Button „Speichern“ und einen SaveDialog. Diesen rufen wir wieder mit Execute auf. Den Inhalt des Memos können wir SaveToFile in den ausgewählten Dateinamen speichern.

Außerdem wollen wir den Text aus einem Edit an den Text im Memo anhängen. Dazu benötigen wir einen Button „Hinzufuegen“ und ein Edit. Mit Memo.Lines.Add kann der

Edit.Text angefügt werden.

### 3.3 Haus vom Nikolaus

In einer PaintBox (unter Additional) soll das Haus vom Nikolaus gezeichnet werden. Dazu muss das Ereignis OnPaint der Paintbox implementiert werden. Mit PaintBox.Canvas.MoveTo bewegt man den Stift und mit LineTo zieht man eine Linie von der letzten Position zur angegebenen Position. Z.b. MoveTo(10,10) und anschließend LineTo(20,20) zieht eine Linie vom Pixel 10,10 nach 20,20. Anschließend kann man weitere Linien ziehen(LineTo) oder den Stift ohne Linie bewegen(MoveTo). Das Haus soll 75 Pixel hoch sein und 50 Breit. Die Paintbox muss also entsprechend größer sein als das.