Minimalanforderungen an die Projektarbeit

- Beschreibung der Aufgabenstellung (Lastenheft)
  - Beschreibung des geplanten Projekts
  - Vorentwurf der Benutzeroberfläche
- Vorentwurf des Klassendiagramms
- Aufteilung des Projekts auf die Gruppenmitglieder
- Terminplanung

Vorentwurf:	4.5.15 (13:00 A301 oder Pausenhalle)
überarb. Vorentwurf:	nach Vereinbarung
Zwischenstand:	8.6.15
Dokumentation:	22.6.15
Präsentation: (ab)	29.6.15

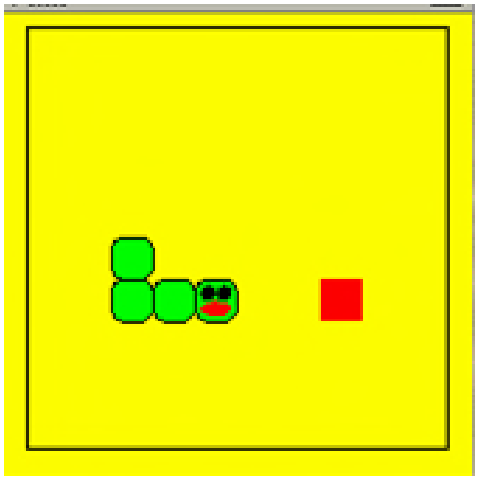
- **Objektorientierter Entwurf**, 3-Schichten-Architektur
- **Schriftliche Dokumentation des Entwurfsprozesses durch Diagramme und Erläuterungen**
  - **Klassendiagramm mit Erläuterungen**
    (Bedeutung der Werte von Attributen und der Rückgabewerten von Operationen)
  - **Beschreibung aller wichtigen Szenarien**
    (Sequenzdiagramme, falls notwendig mit Erläuterungen)
  - **Zustandsdiagramm**
  - **Darstellung aller nicht trivialen Operationen als Struktogramm**

- Verifikation des Entwurf durch Implementierung

- Erstellung einer Bedienungsanleitung

- Präsentation der Projektarbeit
  - Jeder Projektteilnehmer präsentiert seinen (!) Beitrag zum Projekt.

P1: Wurmspiel



Ein Computerspiel mit dem Namen „Wurmspiel“ soll entworfen werden.

Auf den Zellen eines quadratischen Spielfeldes (Display) bewegt sich ein Wurm.

Der Spieler wählt mit den Pfeiltasten die Richtung der Kriechbewegung des Wurms aus.

In einer zufällig ausgewählten Zelle außerhalb des Wurms befindet sich Futter. Findet der Wurm das Futter, so wird er um eine Zelle verlängert und es wird neues Futter ausgelegt.

Trifft der Wurmkopf auf eine Wurmzelle oder auf die Wand, so ist das Spiel beendet, es wird die Meldung „verloren“ ausgegeben.

Hat der Wurm die Endlänge 10 erreicht, so ist das Spiel ebenfalls beendet und es wird die Meldung „gewonnen“ ausgegeben.

Die Anfangslänge des Wurms beträgt vier Zellen. Das Spielfeld ist aus 10 Spalten ($0 \le x \le 9$) und 10 Zeilen ($0 \le y \le 9$) aufgebaut.

Zur Modellierung des Spiels müssen die folgenden Klassen benutzt werden:
Oberflaeche, Steuerung, Wurm, Wurmzelle. (Weitere Klassen sind möglich.)

P2: Computerspiel „Fang die Katze“

Das Spiel „Fang die Katze“ soll als Computerspiel für zwei Spieler entworfen werden.

Das Spiel wird auf einem 8 x 8 Felder großen Spielfeld gespielt. Wenn Spieler 1 am Zug ist, kann er die Katze (K) mit den Tasten ’L’ (nach links), ’R’ (nach rechts), ’O’ (nach oben) und ’U’ (nach unten) jeweils um ein Feld verschieben. Spieler 2 kann bei jedem Spielzug eine Barriere (#) mit der Computermaus an eine beliebige, freie Stelle des Spielfelds setzen.

Spieler 2 hat gewonnen, wenn die Katze in ihrer Bewegungsfreiheit vollständig eingeschränkt ist, d. h. Spieler 1 keinen Zug mehr machen kann. Schafft es Spieler 2 jedoch nicht die Katze mit höchstens N Barrieren in ihrer Bewegungsfreiheit vollständig einzuschränken, dann ist die Katze frei und Spieler 1 hat gewonnen. (N wird vor dem Start des Spiels festgelegt.)

	0	1	2	3	4	5	6	7
0								
1							#	
2								
3				#				
4								
5				*K*				
6			#					
7								

Statuszeile:

Während des Spiels wird in der Statuszeile angezeigt, welcher Spieler gerade am Zug ist. Das Programm reagiert nur auf Eingaben des Spielers, der am Zug ist.

Unzulässige Spielzüge:

Spieler 2 versucht eine Barriere auf ein Feld zu setzen, das schon durch eine Barriere oder die Katze belegt ist.

Spieler 1 versucht die Katze über den Rand hinaus oder auf ein Feld, auf dem eine Barriere steht, zu schieben.

Zur Modellierung des Spiels müssen die folgenden Klassen benutzt werden:
Oberflaeche, Steuerung, VirtuellesSpielfeld, Katze. (Weitere Klassen sind möglich.)

P3: Virtuelles Schachbrett

Tw	Sw	Lw	Dw	Kw	Lw	Sw	Tw
Bw	Bw	Bw	Bw	Bw	Bw	Bw	Bw
Bs	Bs	Bs	Bs	Bs	Bs	Bs	Bs
Ts	Ss	Ls	Ds	Ks	Ls	Ss	Ts

Die Figuren werden durch Anklicken der Figur und durch Anklicken des gewünschten Zielfeldes bewegt.

Das Programm prüft die Zulässigkeit eines Zuges (Farbe am Zug und Zielfeld erreichbar). Außerdem wird geprüft, ob eine Figur geschlagen wird (entfernt werden muss) und ob einer der beiden Könige „schachmatt“ gesetzt wurde.

Zur Modellierung des Spiels müssen die folgenden Klassen benutzt werden: Oberflaeche, Steuerung, VirtuellesSpielbrett und für jede Spielfigurentyp (Turm, Springer, Läufer, …) eine eigene Klasse.

P4: Virtuelles Damespielbrett

	W		W		W		W
W		W		W		W	
	W		W		W		W
S		S		S		S	
	S		S		S		S
S		S		S		S	

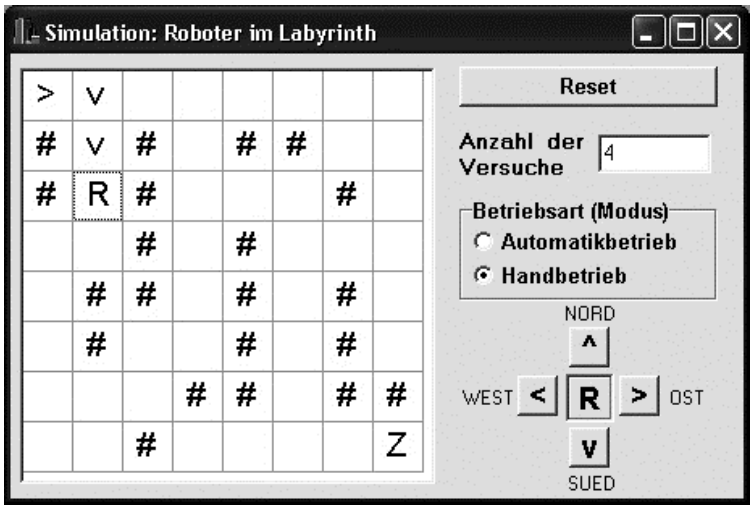
Die Figuren werden durch Anklicken der Figur und durch Anklicken des gewünschten Zielfeldes bewegt.

Das Programm prüft die Zulässigkeit eines Zuges (Farbe am Zug und Zielfeld erreichbar). Außerdem wird geprüft, ob eine Figur geschlagen wird (entfernt werden muss) und ob eine der Dame zugeteilt werden muss.

Zur Modellierung des Spiels müssen die folgenden Klassen benutzt werden: Oberflaeche, Steuerung, VirtuellesSpielbrett und für jede Spielfigurentyp (Knopf, Dame) eine eigene Klasse. (Weitere Klassen sind möglich.)

P5: Simulationsprogramm

Ein Roboter soll selbstständig einen Weg durch ein Labyrinth finden.
Zum Testen des Navigationsalgorithmus soll eine Simulationsumgebung modelliert werden.
Bild 1 zeigt einen Vorschlag für die Benutzeroberfläche der Simulationsumgebung.



← Bild 1


Bild 2.a


Bild 2.b


Bild 2.c


Bild 2.d

Der Roboter kann das 8 x 8 Felder große Labyrinth nicht verlassen. Die Hindernisse innerhalb des Labyrinths sind durch '#'-Zeichen dargestellt.

Die Position des Roboters wird im Ruhezustand durch 'R' dargestellt (Bild 2.a und Bild 2.d**)**.
Bei Fahrversuchen wird die Roboterposition durch einen Pfeil, der in die gewünschte Bewegungsrichtung zeigt, dargestellt (Bild 2.b und Bild 2.c).
Wenn die Bewegung in dieser Richtung möglich war, wird die neue Roboterposition durch ein 'R' gekennzeichnet (Bild 2.d). Der Pfeil bleibt in diesem Falle stehen, um die Bewegungsspur des Roboters zu zeigen.
Kann die Bewegung aber nicht ausgeführt werden (s. Bild 2.c), wird der Pfeil nach einer kurzen Wartezeit wieder durch ein 'R' (s. Bild 2.a) ersetzt.

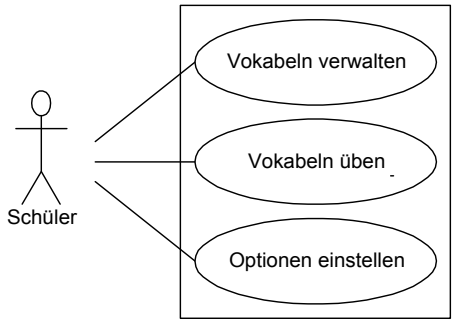
Der Roboter kann im Handbetrieb mithilfe der Pfeiltasten bewegt werden. Wird in den Automatikbetrieb (-> Algorithmus testen) umgeschaltet, dann sucht der Roboter seinen Weg selbst.
Damit die Wegsuche beobachtet werden kann, wird durch einen Timer ein Zeitereignis erzeugt. Bei jedem zweiten Zeitereignis macht der Roboter einen Bewegungsversuch. Für den Beobachter wird das 'R' (aktuelle Roboterposition) durch einen Pfeil in der gewünschten Bewegungsrichtung ersetzt (s. Bild 2.b und 2.c). Beim darauf folgenden Zeitereignis wird die Position des Roboters wieder durch ein 'R' (s. Bild 2.d und 2.a) dargestellt.

Nach dem Starten des Programms bzw. nach dem Drücken der Reset-Taste befindet sich der Roboter in der linken, oberen Ecke. Sein Ziel ist der Punkt 'Z' in der rechten, unteren Ecke. Im Automatikbetrieb bleibt der Roboter nach dem Erreichen des Zielpunktes stehen.
Die Positionen der Hindernisse, die bei erfolglosen Bewegungsversuchen erkannten wurden, speichert der Roboter ab (-> Gedaechtnis).

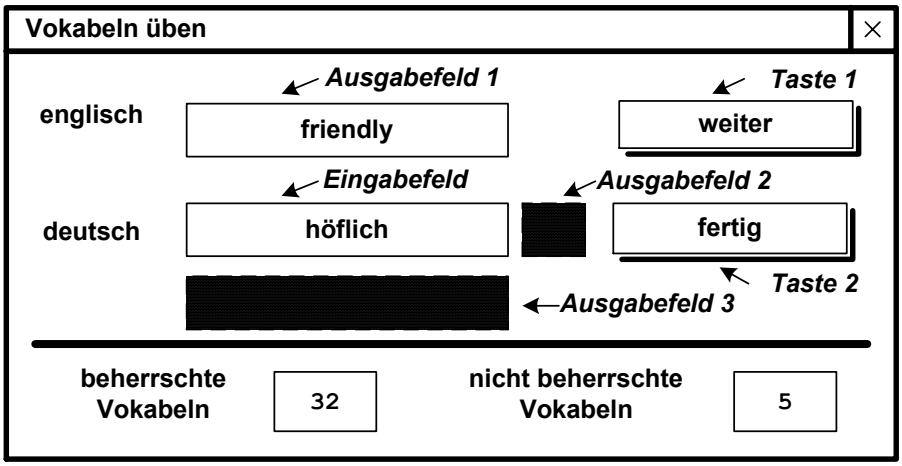
Zur Modellierung des Spiels müssen die folgenden Klassen benutzt werden:
Oberflaeche / Gui, VirtuelleWelt / Labyrinth, Roboter, Gedaechtnis.
(Weitere Klassen sind möglich.)

P6: Vokabeltrainer

Ein Vokabeltrainer soll entworfen werden. Der Vokabeltrainer deckt die drei dargestellten Anwendungsfällen ab.



Die Benutzeroberfläche für den Anwendungsfall „Vokabeln üben“ ist hier beispielhaft dargestellt. Im Ausgabefeld 1 wird eine Vokabel angezeigt. Der Anwender muss seine Antwort in das Eingabefeld schreiben und anschließend die Taste 2 ('fertig') drücken. Im Ausgabefeld 2 wird abhängig davon, ob die Antwort richtig oder falsch ist, die Bewertung „ok“ in grün oder „X“ in rot angezeigt.



Durch das Drücken der Taste 2 ('fertig') ändert sich der Text auf der Taste 2 von „fertig“ auf „Lösung anzeigen“. Der Anwender kann jetzt entscheiden, ob er die korrekte Antwort im Ausgabefeld 3 sehen möchte oder ob er sofort die nächste Vokabel anfordern möchte. Um die Lösung zu sehen, muss er Taste 2: 'Lösung anzeigen' drücken. Wenn er die Taste 1 ('weiter') drückt, wird die nächste Vokabel angezeigt und ein neuer Abfragezyklus beginnt. Auf Taste 2 steht jetzt wieder „fertig“.

Im unteren Teil des Fensters wird der Lernerfolg ausgegeben. Die Anzeige wird immer nach dem Betätigen der Taste 'fertig' aktualisiert. Eine Vokabel wird als „beherrscht“ gewertet, wenn sie dreimal korrekt beantwortet wurde. Bei einer falschen Antwort wird der Zähler wieder auf den Initialisierungswert 0 zurückgesetzt.

P7: Fahrkartenautomat

Eine Automatenfabrik erhält den Auftrag, einen Fahrkartenautomat für eine kleine Bahngesellschaft zu entwickeln. Der Automat soll nach den Vorgaben der Bahngesellschaft die folgende Ausstattung haben (siehe auch Bild 1):

- Eine auf Papier gedruckte **Liste der Haltepunkte** mit Kode-Nummern der Haltepunkte,
- eine **Tastatur** mit den Ziffern „0“ .. „9“ und „A“ zur Eingabe des Fahrziels,
- eine einzeilige, alphanumerische **Anzeige**,
- eine **Datenmodul** zur Speicherung der Fahrziele
- eine **Kasse** mit Geldeinwurf (Münzen und Scheine), Geldspeicher und Geldausgabe,
- einen **Drucker** zum Drucken der Fahrkarte,
- einen **Ausgabeschacht** für die Fahrkarten und das Rückgeld.

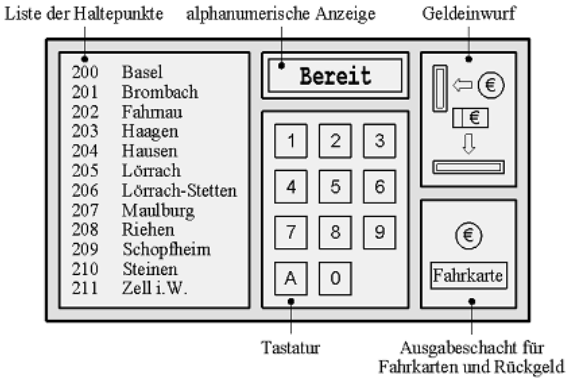


Bild 1: Mögliches Aussehen des Fahrkartenautomaten

Ein Fahrkartenkauf am Automat verläuft wie folgt:

1. Der Fahrkartenautomat zeigt auf seiner Anzeige den Text „Bereit“ an.
2. Der Kunde sucht in der alphabetisch sortierten Liste der Haltepunkte den Ortskode für sein Fahrziel und
3. gibt die Kode-Nummer über die Tastatur ein. Der Abfahrtort ist immer der Standort des Automaten.
4. Nach Eingabe der dritten Ziffer wird geprüft, ob ein Haltepunkt für den eingegebenen Kode existiert.
5. Existiert das Fahrziel, so werden die Daten der Fahrkarte ermittelt, der Fahrpreis wird auf der Anzeige angezeigt.
6. Nun kann der Kunde den angezeigten Betrag einwerfen. Hat der Kunde den Betrag vollständig bezahlt oder aufgrund von fehlendem Kleingeld überzahlt, ist der Fahrkartenverkauf nicht mehr abzubrechen.
7. Der Fahrkartenautomat druckt eine Fahrkarte aus und gibt evtl. Wechselgeld zurück. Der Fahrkartenverkauf ist abgeschlossen. Der Kunde entnimmt die Fahrkarte und evtl. das Wechselgeld.
8. Weiter bei 1.

Bis zu Schritt 6 kann der Kunde durch Betätigen der Abbruch-Taste „A“ auf der Tastatur den Geschäftsvorgang abbrechen. Spezielle Betriebszustände wie z.B. kein Papier im Drucker, kein Wechselgeld oder Stromausfall während des Fahrkartenverkaufs werden bei dieser Aufgabe nicht berücksichtigt.

Ortsname	Basel	Riehen	Lörrach-Stetten	Lörrach	Haagen	Brombach	Steinen	Maulburg	Schopfheim	Fahrnau	Hausen-Raitbach	Zell i.W.
Ortskode	200	208	206	205	203	201	210	207	209	202	204	211
Entfernung[km]	0	5	7	8	10	12	15	19	22	24	26	29
Index	0	1	2	3	4	5	6	7	8	9	10	11

Bild 2: Bahnlinie

Das Bild 2 zeigt die Bahnlinie, für die der Automat Fahrkarten verkaufen soll. Die Bahnlinie besitzt 12 Haltepunkte. Ein Haltepunkt wird durch den Ortsnamen, den Ortskode und eine Entfernungsangabe beschrieben. Die Entfernungsangabe speichert die Entfernung vom Haltepunkt Basel.