Projekt Kalaha – Gesamtdokumentation

Projektleitung: Eberhard Lehmann, Konstantin Hicke, Vera Juhre

Hergestellt vom Informatik-Grundkurs 2. Sem./2000 der Rückert-Oberschule Berlin

Mitwirkende: Mehmet Aydin, Benjamin Böhmer, Céline Danto, Ahmet Erisen, Kirsten Heibey, Konstantin Hicke, Martin Hicke, Vera Juhre, Robert Kaplun, Enric Karkossa y Mora, Christoph Klammt, Dennis Knospe, Thomas Kolonko, Timo Lange, Markus Liwicki, Jorma Marquardt, Martin Noblé, Tobias Schaefer, Kolja Schwenghagen, Jens Winhold

Entwicklungszeitraum: 24.2.2000 – 10.07.2000
Arbeitsstunden: 39 Schulstunden

Inhaltsverzeichnis

Tätigkeiten der Gruppen

Datum	Spielregeln	Menüoberfläche	Spieloberfläche	Spiel Mensch – Mensch	Spiel Mensch – Computer	Bohnenbe-wegung	Lehmann	Projektleitung
13.03.	Konzept	Konzept	Konzept	Konzept	Konzept	Konzept	Präzisierung der Gruppenarbeit	Dateistruktur
16.03.	Dokumentation des Konzeptes	Dokumentation des Konzeptes	Dokumentation des Konzeptes	Dokumentation des Konzeptes	Dokumentation des Konzeptes	Dokumentation des Konzeptes	Hinweis zu Schnittstellen-minimalität	Dateiablegung, Gruppentätig-keiten
20.03.	Struktugramm	Skizzierung	Spielfeldentwurf	Struktugramm	Struktugramm, Dokumentation, Algorithmus	Simulation einer Hand als Mauscursor	Ratschläge	Dokumentation
23.03.	Schnittstellen-planung	Schnittstellen-planung	Schnittstellen-planung	Schnittstellen-planung	Schnittstellen-planung	Schnittstellen-planung	Schnittstellen-planung	Sammlung der Schnittstellen-planung
27.03.	Aufschreiben der Spielregeln mit Word	Entwurf der Menüoberfläche	Muldenbenen-nung	Dokumentation des Algorithmus	Protokoll, Schnittstelle, Testobefläche	Mauscursorkon-figuration	Scnittstellenfest-legug	Dokumentation, Datenstruktur
03.04.	Spielregeln, Struktugramm	Fertigstellung des Menüs	Anklickbarkeit der Mulden	Algorithmus	Arbeit am Programm	Transparenz und Cursor, Drag & Drop	Beratung, entgültige Fassung des Schnittstellen-bogens	Unterstützung der Spielober-flächengruppe, sonstiges
10.04.	Fertigstellung des Struktugramms	Ausarbeitung mit neuem Oberflächebild	Bildbearbeitung	Zusammen-fügung der Units von Testoberfläche und Algorithmus	Kommunikation zwischen Units	Drag & Drop	das Übliche	Organisation, Bildgröße, Hilfestellung

17.04.	1)Struktogramm 2) html-Hilfedatei	Buttons für Menüoberfläche	Kooperation mit Bohnenbe-wegung, Einbindung der html-Hilfe	Programmierung von bestimmten Sonderfällen	**Bis auf kleine Fehler fertig. Schnittstelle: anzahl bohnen**	Koordination mit Oberfläche, Bildbewegung	das Übliche	Organisation Überwachung, Hilfestellung
08.05.	Integration	Integration, Abschluß der Menüoberfläche	Integration der Spielregeln	Algorithmus, Dokumentation, Kommentare	Algorithmus, Dokumentation, Kommentare	Algorithmus	Hilfe bei der Organistation, Material	Hilfestellung an Bohnenbe-wegung und Spielregeln
18.05.	Fertigstellung der Hilfedatei	Verfeinerung der Oberfläche	Zusammenarbeit mit MM	Zusammenarbeit mit SO	endgültiges Abschließen des Programms	Vorbereitung Referat	Hilfe bei der Organistation, Material	nicht anwesend
22.05.	nicht anwesend	Graphik der ein/zwei Spielerauswahl verfeinert	Zusammenfü-gung von MM und Spieloberfläche	Zusammenfü-gung von MM und Spieloberfläche	Weiterentwick-lung der künstlichen Intelligenz	Vorbereitung Referat	Hilfe bei der Organistation, Material	nicht anwesend
29.05.	nicht anwesend	Präsentation des Teilprogramms **mo_dpr_170400 .exe** Integration der beiden Teil-programme zu KALAHA.EXE in „Spieloberfläche“	Integration	Präsentation der Oberfläche Verzeichnis: **sopr.exe** in Spieloberfläche, Integration der beiden Teilprogramme zu KALAHA.EXE in „Spieloberfläche“	Problemlösung (bei der Entwick-lung der künst-lichen Intelligenz traten Probleme auf)	Vorbereitung Referat	Hilfe bei der Organistation, Material	Weiterführung der Dokumentation
19.06.	fertig	Wir sind fertig und Helden!		fertig	fertig	Referat	Organisation	Beratendes
03.07.	Präsention	Präsention	Präsention	Präsention	Präsention	Präsention	Präsention	Präsention

Gruppen:

1. *Spielregeln*: Enric, Marin N.
2. *Menüoberfläche*: Jorma, Kolja, Mehmet
3. *Spieloberfläche*: Tobias, Robert, Céline
4. *Spiel Mensch – Mensch*: Jens, Kirsten, Markus, Benjamin
5. *Spiel Mensch – Computer*: Thomas, Christoph, Dennis
6. *Bohnenbewegung*: Ahmet, Martin H., Timo
7. *Projektleitung*: Konstantin, Vera

Regeln zur Bezeichnung und Ablegung der Dateien:

Gruppenkürzel:

8. *Spielregeln*: SR
9. *Menüoberfläche*: MO
10. *Spieloberfläche*: SO
11. *Spiel Mensch – Mensch*: MM
12. *Spiel Mensch – Computer*: MC
13. *Bohnenbewegung*: BB
14. *Projektleitung*: PL

Dateibezeichnungen:

DPR-Dateien: Gruppenkürzel-dpr.dpr
PAS-Dateien: Gruppenkürzel-pas#.pas
andere Dateien: Gruppenkürzel-xxx#.xxx

Beispiele:	SR-dok.doc	*Gesamtdokumentation der Gruppe*
	SR-pas3-130300.pas	*PAS-Datei 3 - Sicherheitskopie*
	SR-pas5.pas	*PAS-Datei 5*
	MM-bmp2.bmp	*Bilddatei*

Sicherheitskopien: Gruppenkürzel-xxx#-datum.xxx
Im separaten Ordner abspeichern!
Bearbeitung der Orginaldatei, bei Bedarf eine erneute Sicherheitskopie mit aktuellem Datum im Extra-Ordner abspeichern.

Datenstruktur

TYPE TMulden: array [1..14] of integer;
Beispiel:

VAR m: TMulden
Aufrufbeispiel:
M[10]

Fenstergröße:
Die vereinbarte Fenstergröße des Programms ist 800x600 Pixel

Spielregeln - Dokumentation

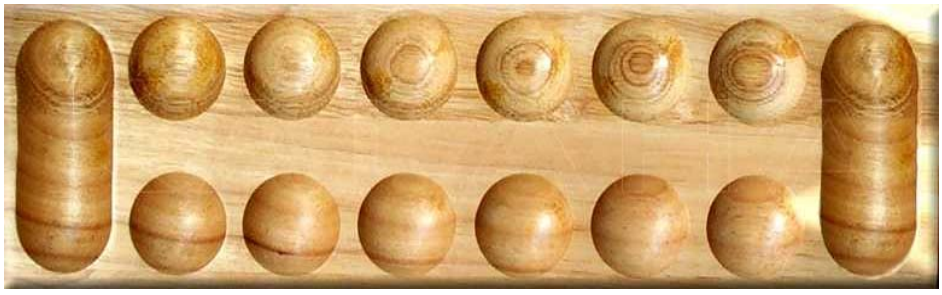
Inhalt

Die Oberfläche

Das Menü

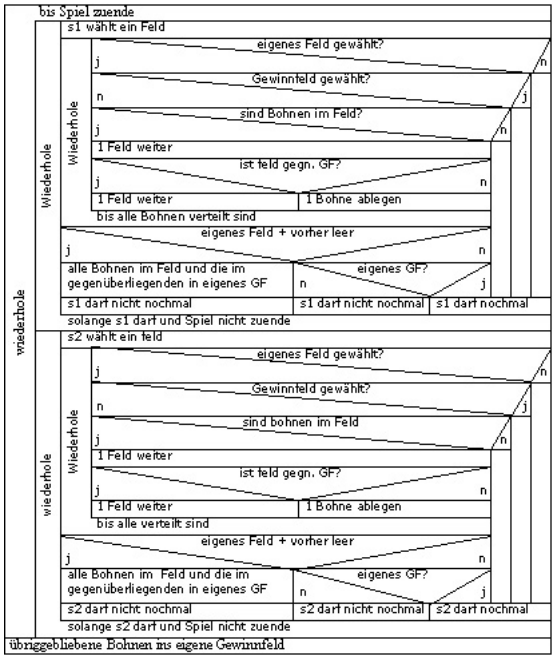
Start:	Startet das Spiel mit entsprechender Spieleranzahl
Ein Spieler:	Mensch vs. Computer
Zwei Spieler:	Mensch vs. Mensch
Setup:	Öffnet weitere Optionen
Info:	Projektinformationen
Hilfe:	Dieses Dokument
Ende:	Beendet das Programm

Das Spielfeld



Die Spielregeln

Das Struktogramm:



Spielablauf

Spielbeginn

1) Auf dem Feld von je 6 Gegenüberliegenden Mulden und je einer Gewinnmulde werden in jede normale Mulde 6 Bohnen verteilt.

2) Der Spieler der Anfängt wird durch Los entschieden.

3) Der erste Spieler wählt nun eine seiner 6 Mulden aus.

4) Alle Bohnen in dieser werden auf die gegen den Uhrzeigersinn folgenden Mulden verteilt.

4.1) Ist die eigene Gewinnmulde unter den folgenden, so wird ebenfalls eine Bohne in dieser platziert.

4.2) Das gegnerische Gewinnfeld wird übersprungen.

4.3) Fällt die letzte Bohne in eine eigene leere Mulde, so kommen alle Bohnen der gegnerischen gegenüberliegenden Mulde ins Gewinnfeld des Spielers.

4.4) Fällt die letzte Bohne ins eigene Gewinnfeld, so darf der Spieler noch mal. Ansonsten fährt der andere genauso wie der erste fort.

5) Das Spiel endet, wenn eine Seite leer ist. Alle verbleibenden Bohnen kommen in das Gewinnfeld des Spielers gelegt, dessen Felder nicht leer sind.

Tipps

1.) Mit 13 Bohnen in einer Mulde landet die letzte Bohne wieder in der jetzt leeren Startmulde. Man erhält die Bohnen des Gegenspielers.

2.) Man kann versuchen, den Gegner "auszuhungern", indem man möglichst wenige Bohnen in seine Mulden legt. So kann erreicht werden, dass er mangels Material das Spiel beenden muss. Hierbei besteht natürlich die Gefahr, dass die eigenen, jetzt gut gefüllten Mulden dem Gegner in die Hände fallen.

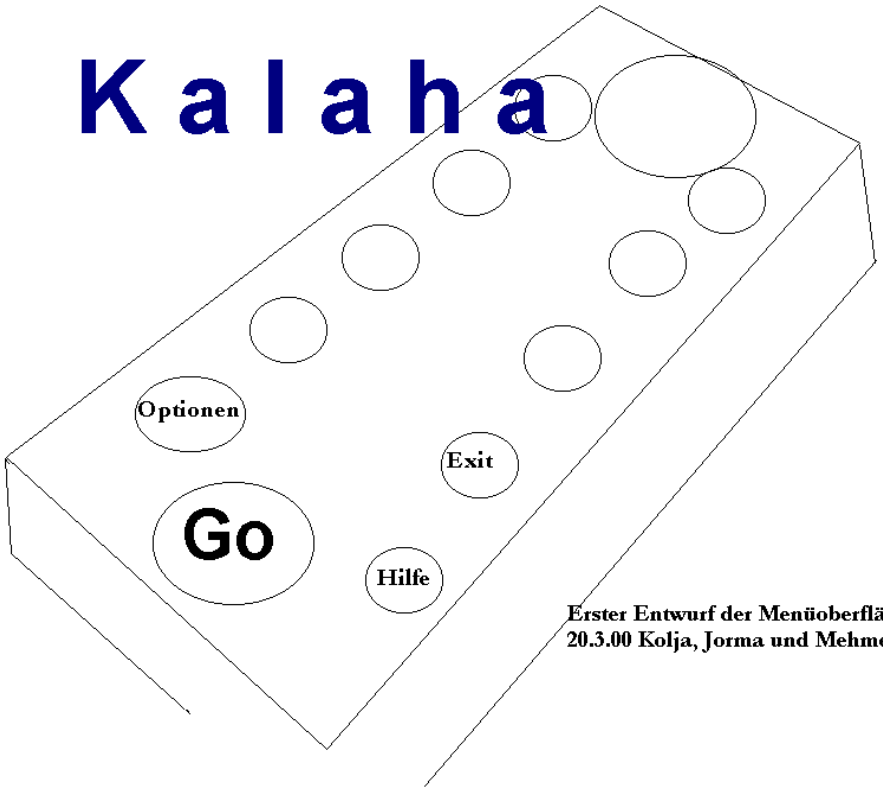
3.) Bei einer Bohne in der letzten Mulde, zwei Bohnen in der vorletzten Mulde, usw. endet der Zug in der eigenen Gewinnmulde und man kann nocheinmal spielen.

Dokumentation Menüoberfläche

14.3.2000
Heute haben wir ein erstes Konzept unserer Menüoberfläche erstellt:
Nach eingehenden Überlegungen und Überprüfung im Gruppengespräch
haben wir uns letztendlich zu folgenden Elementen in unserer Menüoberfläche
durchgerungen:

1. Spielstart-Button zur Auswahl zwischen Spiel Mensch vs. Mensch oder Mensch vs. CPU
2. Options-Button mit Auswahlmöglichkeiten:
   - Steuerung
   - Animationen An/Aus
   - Sound An/Aus
3. Hilfe
4. Beenden

20.3.00



Erster Entwurf der Menüoberfläche
20.3.00 Kolja, Jorma und Mehmet

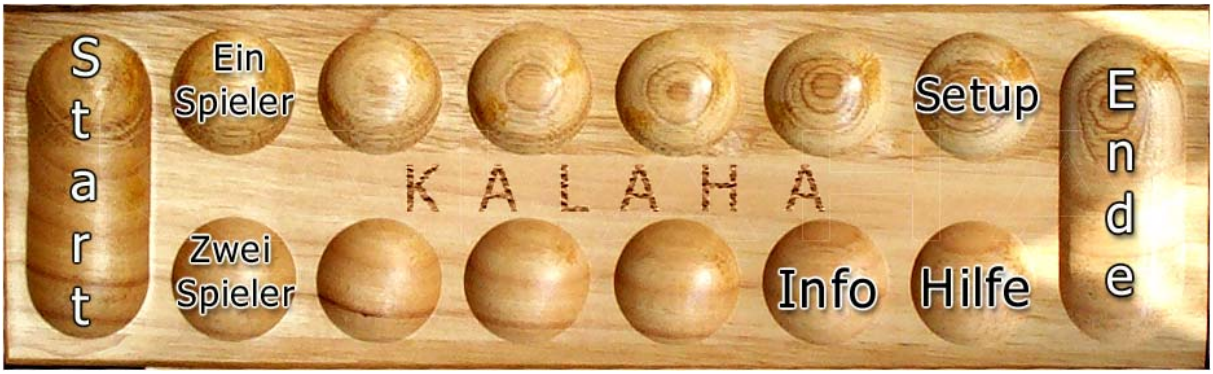
Erstellen einer ersten Skizze mit Paintbrush und Überlegungen zum weiteren Vorgehen.

3.4.00
Erstellen der Menüoberfläche unter Delphi. Festlegung der zu verwendenden Buttons.

10.4.00
Einbinden eines Digitalfotos als Hintergrund für die Menüoberfläche.
Dazu wurde das Spielfeld mit einer Digitalkamera fotografiert und mit einem
entsprechenden Programm bearbeitet.
Das fertige Bild wurde dann in Delphi eingebunden.

08.05.00
Mithilfe der Befehlszeile
shellexecute(handle,'open',Dateipfad\Dateiname,'','',SW_SHOWNORMAL);
werden die Spielregeln im HTML-Format aufgerufen.

18.05.00
Optimierung der Oberfläche, grafische Retuschen
(neue lesbare Schrift in die Oberfläche eingebunden,weil die alte Schrift unleserlich war etc.)



22.05.00
Veränderung der Spielerauswahl
(ausgewählter Spielauswahlbutton leuchtet)

29.05.00
Präsentation des Teilprogramms und Implementierung der anderen Teilprogramme

19.05.00
Mitarbeit an der Integration der verschiedenen Gruppen
Layouten der Dokumentation

Dokumentation Spieloberfläche

Do, 16.03.2000:

1. Ideen:
   - Spielfeld mit Digitalkamera fotografieren und in Delphi einfügen
   - Brett horizontal darstellen
   - Buttons: „Ende"; „Neustart"; „Zurück zum Hauptmenü"
   - Menü-Leiste
   - Message-Fenster
   - Skalieren des Brettes
   - Anzeigen von Gewinner mit Bohnenmenge

Mo, 20.03.2000:

- Beginn der Spieloberflächengestaltung mit Erstellung der Buttons in Delphi 3

Do, 23.03.2000

- Schnittstellenangabe und –beratung mit den anderen Gruppen

Mo, 27.03.2000

- Bezeichnung der Buttons in Delphi 3
- globale Festlegung der Muldenbezeichnung
- Schnittstellenüberlegungen mit der Gruppe „Spielmenü"
- Überlegungen zur Ausgabe von evtl. Fehlermeldungen

Mo, 03.04.2000

- Buttons durch Image-Felder ersetzt

- Menüleiste erstellt
- Variable „angeclickt" erstellt, die den Mulden Nummern zuweist

Mo, 10.04.2000

- digitales Spielbrett graphisch bearbeitet und ins Delphi-Formular importiert.
- Festlegung des Spiel-Fensters auf 800x600 Pixel
- Maße des Bretts: 775x225 Pixel
- Image-Felder in Delphi dem Brett angepaßt

Mo, 08.05.2000

- die Hilfe-Datei muß in das Programm eingefügt werden
- eine html-Datei wird folgendermaßen aufgerufen:
  shellexecute(handle,'open',Dateipfad\Dateiname,'','',SW_SHOWNORMAL);
- folgende Bibliothek muß noch aufgerufen werden: ShellApi
- muß eine exe-Datei aufgerufen werden, sieht der Aufruf so aus:
  winexec('Dateipfad\Dateiname',0)

Mo, 22.05.2000

- Zusammenfügen der Spieloberfläche mit der Mensch-Mensch-Gruppe
- (Prozeduren der Icons kommen aus der MM-Unit)
- Einfügen von Labels für die Bohnenanzahl pro Mulde

Dokumentation der Gruppe Mensch-Mensch

Vom 6.03. – 22.05.00

6.03.2000

Vorhaben:	-Startkonfiguration:	-Spielfeldaufbau -Eingabe der Spielernamen
	-Spielalgorithmus	-Sonderzüge -Fehlervermeidung -Rückstellbutton -Statusbalken
	-Gewinnauswertung	-Auszählung der Bohnen -Anzeige des Gewinners -(Highscore-später)
	-Sonstiges(später)	-multipler Eingabe-Algorithmus

23.03.2000:

Anfang des Erstellens einer **Probeprozedur**, um zu prüfen, ob unser Grundalgorithmus funktioniert.

27.03.2000:

Weiteres Überdenken der Prozedur, um die **Anzahl der Schnittstellen** zu minimieren.
Danach: Schreiben der Unit mit neuem Struktogramm.

03.04.2000

Heute brachten wir unseren **Algorithmus auf den Computer**. Wir haben schon folgende Sonderfälle berücksichtigt: Abwechselndes Zugreifen auf Gewinnmulden, also Fallunterscheidung bei den Spielern.
Nur der Fall, daß man auf einer leeren Mulde endet und somit die Bohnen des gegenüberliegenden Feldes bekommt, ist noch nicht berücksichtigt.
Des Weiteren schrieben wir eine kleine **Testoberfläche** und versuchten weiterhin unsere Schnittstellen zu den anderen Gruppen zu minimieren. Unser Algorithmus ist nunmehr von 3 auswärtigen Variablen abhängig, was schon das Minimum darstellt.
Als nächstes haben wir eine Testoberfläche erstellt, auf der nun schon erste Spiele möglich sind.

10.04.2000

Heute versuchten wir zunächst erfolglos unsere Testoberfläche als eigenständige Unit zu gestalten. Wir scheiterten daran, daß keiner von uns wußte, wie man zwei Units miteinander verbindet.
Zu kämpfen hatten wir zunächst mit ‚**external**' und ‚**forward**' Befehlen, die benötigt werden, um Prozeduren aufzurufen, die sozusagen noch nicht definiert sind. Allerdings stellte sich schnell heraus, daß diese Befehle gar nicht benötigt werden.
Des Weiteren war es noch ein Hindernis, dass gewisse Variablen so deklariert werden, daß sie auch für beide Units gültig sind, sozusagen ‚über-global'.

Anleitung zum Zusammenbringen von Units:

1. Unter ‚**uses**' wird in der zugreifenden Unit der Name der Unit, auf die zugegriffen wird, eingetragen.
   (In unsrem Falle: Die Unit mit der Testoberfläche greift auf die Unit mit unseren Algorithmen zu)

2. In einer Unit werden die nötigen Variablen global deklariert (i, ‚**interface**'-Teil).

```
var
  Form1: TForm1;
  M :TMulden;
  Spieler1:Boolean;
```

3. Die Prozeduren auf die zugegriffen wird, werden in der ‚**Type-Liste**' hinzugefügt.

```
type  TMulden=array[1..14] of integer;
TForm1 = class(TForm)
procedure spielbewegung(Z:Integer);
```

4. Der Aufruf der Prozedur sieht folgendermaßen aus:

```
procedure TForm2.mulde1(Sender: TObject);
begin
Form1.Spielbewegung(1);
Captionveraenderung;
end;
```

8.5.2000

Heute berücksichtigten wir den letzten **Sonderfall (Spielende)** und hatten mit verschiedenen kleinen Algorithmusproblemen zu kämpfen, die wir jedoch lösten. Nun haben wir einen kompletten **Spielzug-Algorithmus**!

22.05.2000

Zusammenfügen der **Spieloberfläche mit der Mensch-Mensch-Gruppe**(Proceduren der Spieloberflächen-Icons kommen aus der MM-UNIT). Einfüge von Labels für die Bohnenanzahl pro Mulde.

Dokumentation Mensch-Computer

Projektbeginn:

Zielsetzung und Bedingungen:

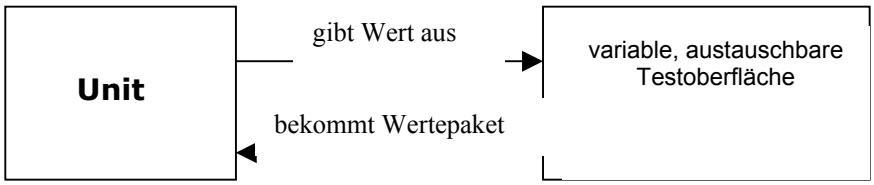
Ziele:
- Eine Computerintelligenz, die gegen einen menschlichen Spieler spielt.
- später: Schwierigkeitsgrade

Bedingungen:
- minimale Schnittstellen: Eine Unit, die über eine Prozedur den Spielstand bekommt und als Rückgabewert die zu ziehende Mulde weitergibt.

Einschränkungen:
- keine graphische Ausgabe
- keine Datenaufnahme außerhalb der Parameterliste. (Keine Datenaufnahme über die Spieloberfläche).

Für die Test- und Entwicklungsphase:

Unsere Unit wird über die gleiche Parameterliste von einem Minimaltestprogramm aufgerufen. Dieses Programm kann einen Spielstand (Benutzereingabe) simulieren und ein Ablaufprotokoll ermöglichen:



Diese Unit kann dann später in das Gesamtprogramm mühelos eingebunden werden.
Stand 16.3.`00

Besprechung der verschiedenen Schnittstellen.
Stand 23.3.`00

-Klärung des Schnittstellenproblems der Gruppen.
-Aufstellen einer ersten allgemeingültigen globalen Datenstruktur:

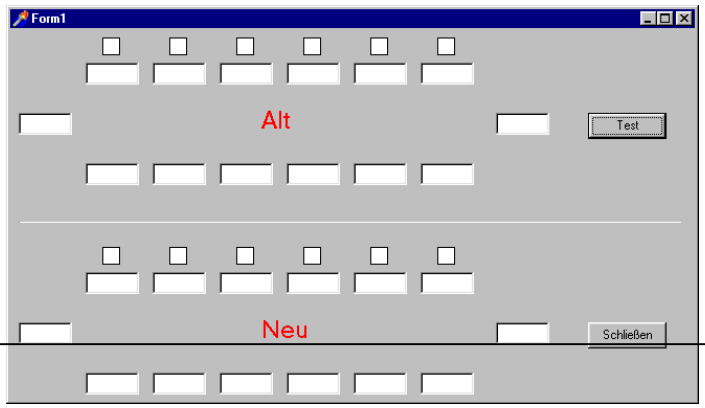
```
Type TMulden = array[1..14] of integer;

Bsp.:
var M :  Mulden;

Aufrufbeispiel:
M[10];
```

Daraus resultierend:

- Die Kommunikation zwischen der Mensch-Computer-Unit und der Testoberfläche wurde daraufhin erweitert
- Die Aufgaben unserer Gruppe wurde um die Kontrolle des Ablaufes und der Reaktion auf Ereignisse in der Spieloberfläche erweitert.

-Algorithmus für primatenhafte künstliche „Intelligenz" (Computer setzt per Zufall)

-Erstellung der ersten simplen Testoberfläche (nur graphisch)



- Diese Testoberfläche hat den Zweck oben unter Alt die Spielsituation anzuzeigen, die der menschliche Mitspieler hinterlässt anzuzeigen. Der Teil Neu zeigt das Spielfeld nach dem Computer gesetzt hat. Man soll der Spielweise des Computers besser dadurch die können.

Stand: 27.3.`00

- Weiterentwicklung der Testoberfläche
  - Dabei wird die Testoberfläche aus programmiertechnischen Gründen auf die Anzeige des aktuellen Spielfeldes beschränkt.
- Entwicklung der grundliegenden Prozeduren (Startverteilung, usw.)
- Entwicklung der Spielalgorithmen
- Programmierung der oben genannten Prozeduren
- Klärung existentieller Fragen
- Programmieren der Interfaces

Stand: 03.04.00

-Weiterentwicklung der Algorithmen für den Spielablauf
-letzte Bohne des Spielers landet in eigener leerer Mulde
-letzte Bohne des Spielers landet in gegnerischer leerer Mulde

-letzte Bohne des Spielers landet in Endmulde
-letzte Bohne des Spielers landet nicht in leerer Mulde
-Algorithmen für die Verarbeitung des Spielzuges
-Kommunikation zwischen den diversen Units
Stand: 10.04.00

-hauptsächlich programmieren und abschließen des ersten einfachen Programms.
-Testen in der Oberfläche und prüfen des Algorithmus
-noch einige kleinere aufgetretene Fehler korrigieren
-Überlegungen einer besseren Computerintelligenz (KI)
-verschiedene Schwierigkeitstufen?
-Programmiertext noch verständlicher machen (andere Variablennamen, logische Bezeichnungen der Prozeduren, Kommentar, etc.)
-Änderung der Testoberfläche
Stand: 08.05.00

-Kommentar zum Quelltext hinzufügen
-Es waren doch noch mehr Fehler zu korrigieren als erwartet => Testen und korrigieren des Programms
-Wer hat zum Schluß gewonnen? => angefangen Gewinnausgabe zu programmieren
Stand: 11.05.00

-endgültiges Beenden des Programms mit einfacher künstlicher Intelligenz
-endgültiges Beenden der Kommentare zum Programm
-Programmtests von außenstehenden Personen durchführen lassen
-keine weiteren Probleme sind aufgetreten
Stand: 18.05.00

-bessere künstlich Intelligenz entwickelt:
- der Computer schaut, ob er 13 Steine in seine eigene Mulde besitzt, damit er dann den letzten Stein in sein eigenes Feld setzen kann um auch die gegenüberliegenden gegnerischen Steine zu bekommen.
- der Computer prüft ob er die Chance hat, sein letzten Stein in seine eigene Gewinnmulde zu legen, damit er nochmals an die Reihe kommt

-Kommentar zu des neu programmierten einfügen
-kleinere Fehler sind aufgetreten, weil der Computer nachdem er sein letzten Stein in seine Gewinnmulde setzte , nicht noch mal an der Reihe war.
-sonst lief alles einwandfrei und ohne Probleme
Stand: 22.05.00

Nach dem die funktionsfähige Version V3 gesichert wurde, arbeiten wir an der Verbesserung der Intelligenz. Die Testoberfläche wurde bei der Integration in die Spieloberfläche abgekapselt und nur die Uni Test_pas_MC.pas in die Spieloberfläche übernommen
Stand: 19.06.00

Dokumentation Bohnenbewegung

16.03.2000

Ergebnisse des Tages: Als erstes überlegten wir uns, wie man die Bohnenbewegung am besten realisieren könnte. Wir einigten uns auf eine Animation unter Zuhilfenahme eines Bildes einer menschlichen Hand. Dies wird auf Grund der relativ wenigen Animationsphasen bevorzugt. Ziel ist es den Mauszeiger als Hand darzustellen. Dieser wird dann in der Animationsphase vom Computer gesteuert.

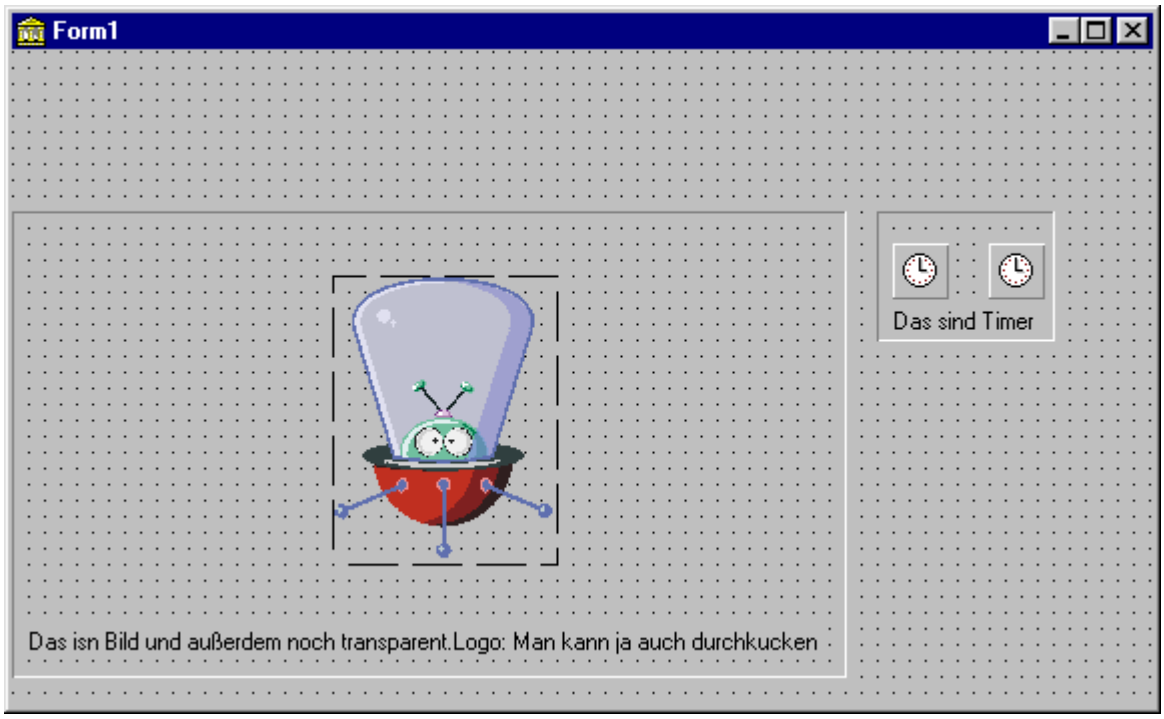
03.04.2000

Wir verwarfen das anfänglich favorisierte Vorhaben, den Cursor als Hand darzustellen, da die Cursorgröße auf 32*32 Pixel beschränkt ist. Uns kamen einige neue Ideen wie man die Animation verwirklichen könnte:
Mit Hilfe eines unsichtbaren Cursors und des Drag&Drop-Systems sollen Bilder bewegt werden.

Ein anderer Vorschlag war, daß einfach das Bild auf die Koordinaten bewegt werden soll, indem man unter Zuhilfenahme eines Timers und den Eigenschaften im Objektinspektor die entsprechenden Anweisungen gibt. Dabei entstand das Problem, daß das Bild während der Bewegung „aufflackerte" und somit die Bewegung nicht sonderlich schön aussah.

10.4.2000

Das Drag&Drop-System erweist sich als schwierig: der Bildschirm wird durch ein spezielles Objekt in Spalten und Zeilen eingeteilt. Eine minimale automatische Bewegung unter Zuhilfenahme der Eigenschaften image.top und image.left sowie eines Timers ist bereits möglich.



08.05.2000

Das Bild wird immer vom Mulde zu Mulde verschoben (96 Pixel). Im Normalfall sind dies nur die Spielmulden, also nach links oder nach rechts. Bei den Gewinnmulden nach oben bzw. unten. Das Bild wird alle 30/1000stel Sekunden um einen bestimmten Wert verschoben. Nun besteht das Problem, daß die Bewegung an der entsprechenden Mulde wieder gestoppt werden muß. Mit einem zusätzlichen Zähler soll gewährleistet werden, daß nach einer bestimmten Zeit die Bewegung stoppt. Schwierig ist dabei, daß dann die Timer-Prozedur unterbrochen werden muß, was noch nicht gewährleistet ist.

05.06.2000

Eine Richtungsänderung ist durch die Verschachtelung mehrer Timer möglich. Natürlich gilt das auch für das Anhalten einer Bewegung. Durch eine Abfrage nach der Position des Objekts läßt sich der aktuelle Timer ausschalten oder disablen und ein zweiter aktivieren, wodurch der zweite eine neue Bewegung ausführen kann.

10.07.2000

Leider konnten wir unsere Arbeit auf Grund mangelnder Zeit nicht beenden. Wir beschränken uns darauf die von uns gewonnenen Kenntnisse den anderen Schülern mitzuteilen. Dies wird in Form eines kurzen Referats geschehen, welches wir in den letzten Stunden vorbereiteten.

Dokumentation der Modulintegration

Integration der einzelnen Units in einen Basisquelltext (Menüoberfläche)
Einbindung durch

uses [einzelne Projektunits] + diverse andere von delphi benötigte.

Zuerst werden die **Unit mmpas** (mensch-mensch) in das **Projekt so.dpr** (Spieloberfläche) eingefügt,
d.h. in der **uses-Liste** mmpas hinzugefügt und bei Spielbeginn die Prozedur form1.formcreate (setzt die Mulden auf 6) aufgerufen.
Der Aufruf der Zugprozedur lautet **form1.spielbewegung** (Mulde auf die geklickt wurde)
Dann bestand das Problem mit der Bohnenbewegung, so daß labels in die Spieloberfläche eingefügt werden mußten (Darstellung mit Zahlen). Die Aktualisierung erfolgt durch die neue Prozedur captionveraenderung
[Hinzufügung von Quelltext in die jeweiligen einzelnen Projkektgruppenunits]
Danach wurden die **Unit so.pas in das Projekt der Menüoberfläche eingefügt**,indem ‚so' in die Uses-Liste hinzugefügt wurden. Der Aufruf wird beim Klicken auf das Startimage durchgeführt, indem form2.formcreate (Form von der Spieloberfläche) aufgerufen wird. Als Extra-Feature wurde ein Label kreiert, welches den aktuellen Spielverlauf kommentiert und am Ende den Gewinner anzeigt.

Integration der Mensch-Computergruppe:
Es wurde die Unit test_mc_pas.pas in die **Projektverwaltung eingefügt** und **unter uses eingebunden.** Für die „click-actions" der Spielmulden wurde die **Prozedur mc mit der entsprechenden Muldenzahl aufgerufen.** Nachdem noch eine **Ergänzung für das Spielende** gemacht wurde war die Unit Mensch-Computer integriert.