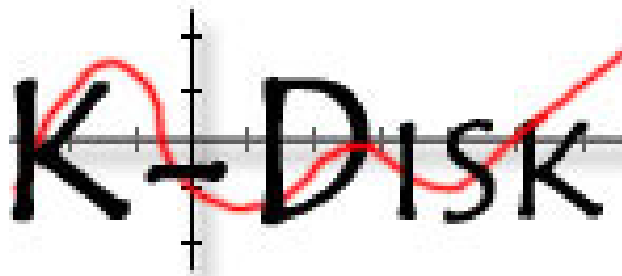


**JAVASTARS**  
SUN MICROSYSTEMS AWARD

**2006**



**Ob zu Hause, in der Schule oder auf der Arbeit, K-Disk berechnet und erklärt Ihnen jederzeit die einzelnen Schritte einer Kurvendiskussion im Detail, sodass Rechenwege nachvollzogen werden können.**

**Dann klappt's auch mit dem Mathelehrer!**

## Projekttitle: K-Disk

### Übersicht

<b>Bundesland:</b>	Nordrhein-Westfalen
<b>Teamnummer:</b>	226
<b>Schulnummer:</b>	
<b>Schulname:</b>	Börde-Berufskolleg Soest
<b>Schulform:</b>	Berufskolleg
<b>Name des Teams:</b>	GWAIN (Group without an interesting name)
<b>Projektname:</b>	K-Disk
<b>Projektkurzbeschreibung</b> (max. 4 Zeilen):	K-Disk ist ein Programm für Mobiltelefone, mit dem man verschiedene Merkmale der Kurvendiskussion errechnen und die Rechenschritte anzeigen kann
<b>Unterrichtsfach:</b>	Anwendungsentwicklung
<b>Gruppengröße:</b>	4
<b>Alter der Schüler/innen:</b>	20-25
<b>Teilnehmende Mädchen:</b>	1
<b>Teilnehmer:</b>	Andreas Sauer andreas.sauer@gmxpro.de  Jean-Pierre Rüth jp.rueth@arcor.de  Rene Starkemeier rstarkemeier@web.de  Janine Gerke janine-gerke@lycos.de

# Inhaltsverzeichnis

<b>1</b>	<b>Allgemeine Fragen zum Wettbewerb</b>	<b>4</b>
1.1	Wie haben Sie/habt ihr von dem Wettbewerb "JAVA STARS 2006" erfahren?	4
1.2	Habt Ihr an anderen Wettbewerben teilgenommen?	4
1.3	Bei welchen Wettbewerben habt Ihr bisher das Programm schon vorgelegt?	4
<b>2</b>	<b>Projektbeschreibung</b>	<b>5</b>
2.1	Projektbeschreibung	5
2.2	Projektidee	5
2.3	Thema	5
2.4	Unterrichtsfach	5
2.5	Nutzen für den Unterricht	5
2.6	Zeitaufwand	5
2.7	Arbeitsumgebung	5
2.8	Java-Applets	5
2.9	Testumgebung	6
2.10	Installation / Start des Programms	6
<b>3</b>	<b>Lösungskonzept</b>	<b>7</b>
3.1	Aufbau der Lösung	7
3.2	Eingesetzte Verfahren	7
<b>4</b>	<b>Programm-Architektur</b>	<b>8</b>
4.1	Übersicht	8
4.2	Funktionskomplex/Klasse 1: ...	8
4.2.1	Zustände	8
4.2.2	Methoden	8
4.3	Funktionskomplex/Klasse 2: ...	8
4.3.1	Zustände	8
4.3.2	Methoden	8
4.4	Funktionskomplex/Klasse 3: ...	8
4.4.1	Zustände	8
4.4.2	Methoden	8
4.5	Funktionskomplex/Klasse 4: ...	8
4.5.1	Zustände	8
4.5.2	Methoden	9
<b>5</b>	<b>Benutzerschnittstelle</b>	<b>10</b>
5.1	Konzept	10
5.2	Funktion 1	10
5.3	Funktion 2	10
5.4	Funktion 3	10
5.5	Funktion 4	10
5.6	Funktion 5	10
<b>6</b>	<b>Referenzen</b>	<b>11</b>
<b>7</b>	<b>Anlagen</b>	<b>12</b>
7.1	Dokumentierter Quellcode	12
7.2	Eidesstattliche Erklärung	12
7.3	Einverständniserklärung der Erziehungsberechtigten	12

# **1 Allgemeine Fragen zum Wettbewerb**

## **1.1 Wie haben Sie/habt ihr von dem Wettbewerb "JAVA STARS 2006" erfahren?**

Unser Lehrer im Unterrichtsfach Anwendungsentwicklung hat uns auf den Wettbewerb hingewiesen.

## **1.2 Habt Ihr an anderen Wettbewerben teilgenommen?**

Nein haben wir bisher nicht.

## **1.3 Bei welchen Wettbewerben habt Ihr bisher das Programm schon vorgelegt?**

Bei keinem.

# **2 Projektbeschreibung**

## **2.1 Projektbeschreibung**

K-Disk ist ein Programm, mit dem man einige Merkmale der Kurvendiskussion errechnen kann. Zu diesen lassen sich dann die einzelnen Rechenschritte anzeigen, sodass der Rechenweg nachverfolgt werden kann.

## **2.2 Projektidee**

Das Projekt ist aus der Idee des Lehrers und der Schüler entstanden. Bei modernen Mobiltelefonen liegen Ressourcen brach, die die Kapazitäten der allermeisten Taschenrechner um ein vielfaches überschreiten. Mit unserem Programm wollten wir den Anfang machen, diese im Unterricht sinnvoll nutzen zu können.

## **2.3 Thema**

Mathematik; mobile Anwendungen.

## **2.4 Unterrichtsfach**

Anwendungsentwicklung.

## 2.5 **Nutzen für den Unterricht**

Wie unter 2.1 bereits erwähnt war die Intention des Programms, die Fähigkeiten moderner Mobiltelefone im Unterricht sinnvoll einzusetzen. Da das Programm die einzelnen Rechenschritte zur Lösung anzeigt und erläutert, leistet es damit eine Ergänzung, die auch mit wissenschaftlichen Taschenrechnern nicht erreicht werden können.

## 2.6 **Zeitaufwand**

Neben einigen Treffen außerhalb der Unterrichtszeit wurden wir während des Fachs Anwendungsentwicklung im Zeitraum von Oktober 2006 bis Januar 2007 für wöchentlich 2 Stunden freigestellt. Insgesamt haben wir also 16 Schulstunden und pro Schüler etwa 8 weitere Stunden aufgewendet.

## 2.7 **Arbeitsteilung**

Janine Gerke und René Stakemeier wendeten sich der Programmierung der einzelnen mathematischen Funktionen zu, die zunächst unter BlueJ entwickelt wurden.

Jean-Pierre Rüth kümmerte sich um die Umsetzung des Projektes unter einer Entwicklungsumgebung, mit der das Programm für mobile Geräte portiert wurde. Dafür wählte er Netbeans mit dem AddOn MobilityPack.

Janine Gerke und Andreas Sauer erstellten die GUI unter Netbeans.

Die Dokumentation wurde von allen Mitgliedern zusammen verfasst.

## 2.8 **Arbeitsumgebung**

Die Entwicklung fand im NetBeans 5.0 mit dem Mobility Pack 7.2 statt, als Zielplattform waren mobile Geräte geplant.

Die Version unseres JSDKs war 1.5.10

## 2.9 **Java-Applets**

Entfällt. Das Programm ist kein Applet.

## **2.10 Testumgebung**

Als Testumgebung kam in der Hauptsache der Emulator des NetBeans Mobility Pack zum Einsatz. Einige Builds wurden aber auch auf unseren Mobiltelefonen getestet. Des Weiteren haben wir uns eine VMWare mit Windows XP und den benötigten Javakomponenten erstellt, so hatten wir eine einheitliche Entwicklungsumgebung und untereinander keine Kompatibilitäts-Probleme.

## **2.11 Probeläufe**

Kommentierte kurz Probeläufe des Programms, die die Lösung der Aufgabe und das Funktionieren des Programms verdeutlichen (erwartete Eingaben, Abläufe, erscheinende Ausgaben, etc.).

## **2.12 Installation / Start des Programms**

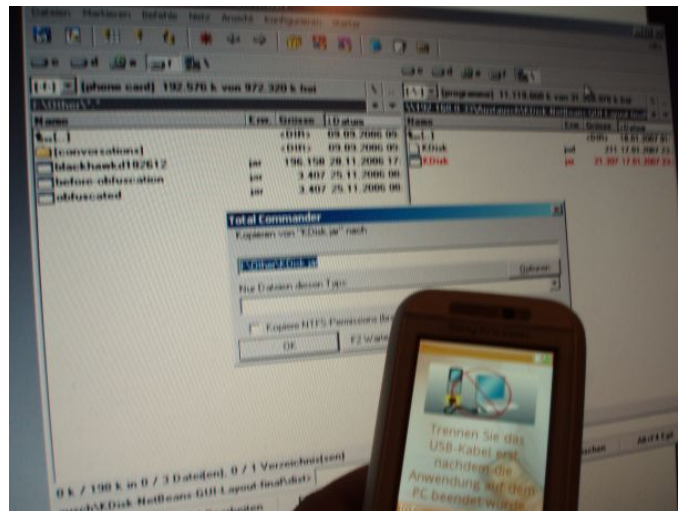
Das Programm liegt in der fertigen Version als .jar-Archiv vor und wird wie für mobile Anwendungen üblich per Bluetooth oder seriell über Kabel oder direkt auf einen MMC/SD-Kartenspeicher übertragen. Je nach Typ des mobilen Gerätes wird dann das Programm dort in einem Programmmenü angezeigt und kann dort gestartet werden.

## Installations- Guide

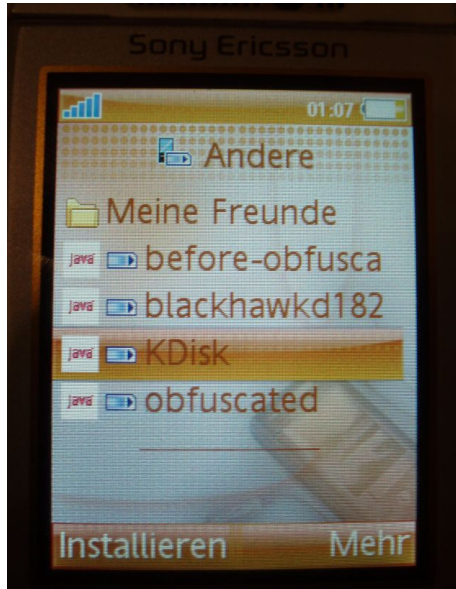
- A) Stellen Sie eine Verbindung zwischen Ihrem Handy und Ihrem Computer her. Weitere Informationen entnehmen Sie dem Benutzerhandbuch des Handys.



- B) Steht eine Verbindung zwischen Handy und Computer, können Sie die KDisk.jar Datei zum Handy übertragen.



C) Nach dem Transfer, müssen Sie K-Disk auf Ihrem Handy installieren



D) Nach der erfolgreichen Installation kann K-Disk gestartet werden und Sie können von nun an die gesamte Funktionalität von K-Disk nutzen



## 3 Lösungskonzept

### 3.1 Aufbau der Lösung

- Brainstorming: welche Funktionen sollen zur Verfügung stehen?  
welche Klassen werden benötigt?  
welche Entwicklungsumgebung wird benutzt?  
Namenskonventionalität von Klassen Methoden und Variablen?
- es wurden folgende Vereinbarungen getroffen (entsprechen der üblichen Java-Nomenklatur)
  - Klassennamen:
    - Ableitung
    - BuildFunction
    - Nullstellen
    - Parser
    - Polinomdivision
    - PqFormel
    - KDisk (Main)
  - Methodennamen werden klein geschrieben und ohne "\_". Jedes neue Wort wird mit einem Großbuchstaben begonnen
  - Klassennamen werden groß geschrieben und ohne "\_". Jedes neue Wort wird mit einem Großbuchstaben begonnen
  - Wie ist der Ablauf des Programms:
    1. Funktion eingeben über ein Textfeld (wird als String eingelesen)
    2. Funktion erkennen (parsen)
    3. Auswahl der Funktionen (Nullstellen, Ableitung...) über Checkboxes
    4. Berechnung
    5. Ausgabe

- Wie sieht das Layout der GUI aus:
  1. Seite:
    - Logo zentriert + Copyright
  2. Seite:
    - Eingabe der Funktion
    - Button: "Nur Ergebnisse der Kurvendiskussion "
    - Button: "inkl. Formeln der Kurvendiskussion "
  3. Seite:
    - Auswahl der Funktionen (Checkboxen):
  4. Seite:
    - Ausgangsfunktion
    - Lösung
- erste Entwicklung der Klassen unter BlueJ
- Umsetzung des Programms in NetBeans - MobilityPack
- Debugging Phase

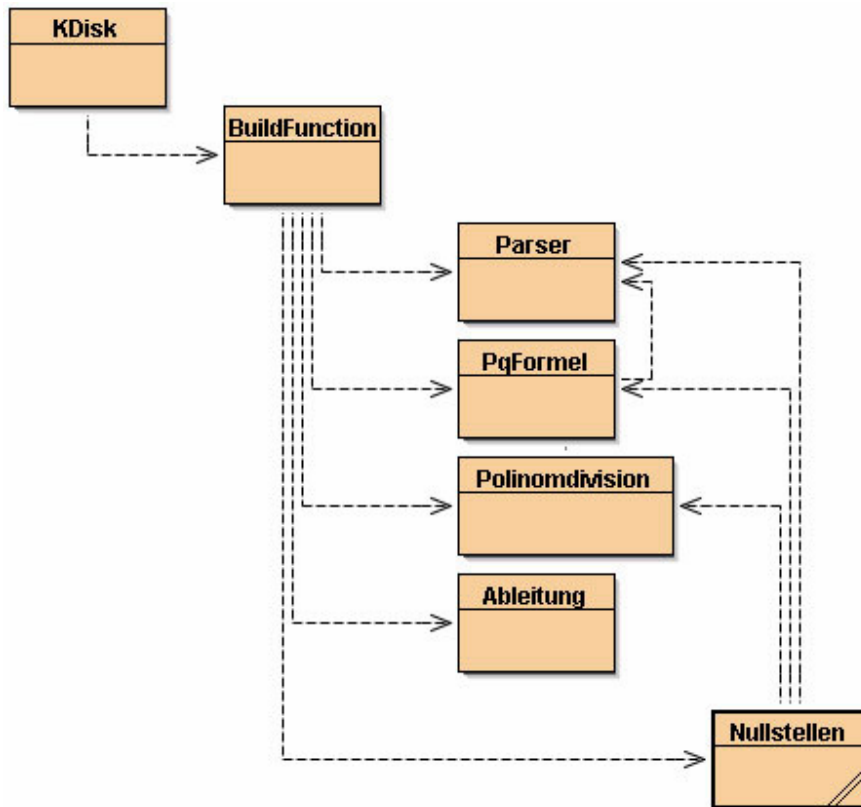
### 3.2 ***Eingesetzte Verfahren***

Extreme Programming

## 4 Programm-Architektur

### 4.1 Übersicht

Aufbau der Klasse und deren Zugriffe:



Die Klasse KDisk ist die Haupt-Klasse und steuert die einzelnen Zugriffe des Objektes "Funktion".

Von der Klasse "BuildFunction" wird eine Instanz erzeugt die verschiedene Attribute und Methoden besitzt.

Die einzelnen Klassen werden unter 4.2 genauer erklärt.

## 4.2 **Funktionskomplex/Klasse 1:KDisk**

KDisk fungiert als Mainklasse; über sie werden alle Funktionalitäten(Ableitung, PgFormel etc) aus den anderen Klassen gehändelt.

### 4.2.1 **Zustände**

BuildFunction func = new BuildFunction()

--> Instanz der Klasse BuildFunction

String strEingabeFunktion;

-->enthält die Eingabefunktion

String strErgebnisString;

-->enthält den ErgebnisString

String strDivisor;

-->enthält den Divisor für die Polynomdivision

int iGradAbleitung;

-->enthält den Grad der Ableitung

boolean zwischenschritte;

-->gibt an, ob Zwischenschritte angezeigt werden sollen

### 4.2.2 **Methoden**

Von den vorgeschriebenen Methoden der Komponenten von KDisk haben wir die Folgenden erweitert:

void commandAction(Command command, Displayable displayable)

Form get\_frmAuswahl()

Form get\_frmLoesung()

TextField get\_txtLoesung()

TextField get\_txtAusgangsfunktion()

get\_frmOptional()

### 4.3 Funktionskomplex/Klasse 2: BuildFunction

BuildFunction ist eine Klasse, von der Objekte erstellt werden können. Die Eigenschaften und Methoden der Klasse entsprechen dabei

BuildFunction
+strEingabe : String
+strErgebnisString : String
+aStringFunktion : String
+aDoubleFunktion : Double
+aPqWerte : Double
+strPqFormel : String
+fehler : String
+dPoliFunktion : Double
+strDivisor : String
+aNullstellen : Double
+printFunktion()
+setEingabe(ein/aus strFunktion : String)
+parse()
+printPqFormel()
+printPqWerte()
+setDivisor(ein/aus strEingabe : String)
+printPolinom(ein/aus zwischenschritte : Boolean)
+printAbleitung(ein/aus zwischenschritte : Boolean, ein/aus iGradAbleitung : Integer)
+printNullstellen(ein/aus zwischenschritte : Boolean)
+getErgebnisString()

#### 4.3.1 Zustände

String strEingabe

--> nimmt die Funktion per Eingabe als String entgegen

String strErgebnisString

--> enthält das Ergebnis für die eingegebene Funktion

String aStringFunktion[]

--> enthält die einzelnen gesplitteten Elemente der Eingabefunktion (Wert, Vorzeichen, Exponent)

double aDoubleFunktion[]

--> enthält die einzelnen gesplitteten Elemente der Eingabefunktion (Wert, Vorzeichen, Exponent)

### Attribute für PQ-Formel

double aPqWerte[]

--> enthält die Ergebniswerte für die PQ-Formel

String strPqFormel

--> enthält die PQ-Formel als String

boolean fehler

--> gibt an, ob bei der PQ-Formel ein Fehler aufgetreten ist

### Attribute für Polinomdivision

double dPoliFunction[]

--> enthält das Ergebnis der Polinomdivision

String strDivisor

--> enthält den Divisor für die Polinomdivision

### Attribute für Nullstellen

double aNullstellen[]

--> enthält alle gefundenen Nullstellen

#### 4.3.2 Methoden

Name	: printFunktion
Funktion	: gibt die Ausgangsfunktion aus
Übergabeparameter	: -
Rückgabewert	: -

Name	: setEingabe
Funktion	: speichert die Grundeingaben für die Funktion (wie den EingabeString) und für die pq-Formel
Übergabeparameter	: String : strFunktion Eingabestring
Rückgabewert	: -

Name	: parse
Funktion	: splittet die Eingegebene Funktion in ihre einzelnen Elemente auf (Wert, Vorzeichen, Exponent)
Übergabeparameter	: -
Rückgabewert	: -

Name	: printPqFormel
Funktion	: Ausgabe der pqFormel als String
Übergabeparameter	: -
Rückgabewert	: -

Name	: printPqWerte
Funktion	: Ausgabe der Werte, die sich aus der pqFormel ergeben
Übergabeparameter	: -
Rückgabewert	: -

Name	: setDivisor
Funktion	: Füllt den Parameter strDivisor
Übergabeparameter	: String: strEingabe   enthält den Divisor als String
Rückgabewert	: -

Name	: printPolinom
Funktion	: Ausgabe des Arrays, das sich aus der Polinomdivision ergibt
Übergabeparameter	: boolean: zwischenschritte   gibt an, ob bei der Ausgabe Zwischenschritte angezeigt werden sollen
Rückgabewert	: -

Name	: printAbleitung
Funktion	: Ausgabe der Ableitung
Übergabeparameter	: boolean: zwischenschritte   gibt an, ob bei der Ausgabe Zwischenschritte angezeigt werden sollen int iGradAbleitung   gibt den Grad der Ableitung an
Rückgabewert	:-

Name	: printNullstellen
Funktion	: Ausgabe der Nullstellen
Übergabeparameter	: boolean: zwischenschritte   gibt an, ob bei der Ausgabe Zwischenschritte angezeigt werden sollen
Rückgabewert	: -

Name	: getErgebnisString
Funktion	: gibt den ErgebnisString zurück
Übergabeparameter	: -
Rückgabewert	: -

#### 4.4 Funktionskomplex/Klasse 3: Parser

Konvertierung der Eingabe der Funktion in die einzelnen Elemente

- Wert
- Exponent

Parser
+strFehler : String
+parsenInStringArray(ein/aus strEingabe : String) : String
+parsenInDoubleArray(ein/aus strArray : String) : Double
+parsenInDoubleArray(ein/aus strEingabe : String) : Double
+findInString(ein/aus strSuche : String, ein/aus delimiter : String) : Boolean
+arrayVergroessen(ein/aus array : String) : String
+arrayVerkleinern(ein/aus array : String) : String
+printArray(ein/aus array : String, ein/aus name : String)
+parsenInString(ein/aus strArray : String, ein/aus fx : Boolean) : String
+parsenInString(ein/aus dArray : Double, ein/aus fx : Boolean) : String
+split(ein/aus aSource : String, ein/aus aDelimiter : Char) : String

##### 4.4.1 Zustände

String strFehler

--> enthält den Fehler Text

##### 4.4.2 Methoden

Name	: parsenInStringArray
Funktion	: splittet die Eingegebene Funktion in ihre einzelnen Elemente auf (Wert, Vorzeichen, Exponent)
Übergabeparameter	: String: strEingabe   der EingabeString, der die Funktion enthält
Rückgabewert	: String[]: aElemente[]   enthält die einzelnen gesplitteten Elemente

Name	: parsenInDoubleArray
Funktion	: wandelt ein Array mit Stringwerten in Double um
Übergabeparameter	: String[]: strArray as EingabeArray, der die einzelnen Elemente der Funktionen enthält
Rückgabewert	: double[]: dArray[] enthält die einzelnen gesplitteten Elemente in double

Name	: <code>parseInDoubleArray</code>
Funktion	: wandelt ein String in ein Array mit Double-Werten um
Übergabeparameter	: <code>String: strEingabe</code>   EingabeString der Funktion
Rückgabewert	: <code>double[]: dArray[]</code>   enthält die einzelnen gesplitteten Elemente in double

Name	: <code>findInString</code>
Funktion	: sucht in einem String nach dem Buchstaben "x"
Übergabeparameter	: <code>String: strSuche</code> enthält den zu durchsuchenden String
	: <code>String: delimiter</code> enthält den Trenner
Rückgabewert	: <code>boolean</code>

Name	: <code>arrayVergroessern</code>
Funktion	: Vergrößert ein Array um 1 Element
Übergabeparameter	: <code>String[]: array</code>   das zu vergrößerte Array
Rückgabewert	: <code>String[]: tmp</code>   das vergrößerte Array

Name	: <code>arrayVerkleinern</code>
Funktion	: Verkleinert ein Array um 1 Element
Übergabeparameter	: <code>String[]: array</code>   das zu vergrößerte Array
Rückgabewert	: <code>String[]: tmp</code>   das vergrößerte Array

Name	: <code>printArray</code>
Funktion	: gibt alle Elemente des Arrays auf dem Bildschirm aus
Übergabeparameter	: <code>String[]: array</code>   das auszugebende Array
	: <code>String: name</code>   Name des Arrays
Rückgabewert	: -



Name	: parsenInString
Funktion	: wandelt die ArrayElemente in einen String um, der die Funktion enthält
Übergabeparameter	: String[]: strArray   enthält die einzelnen Elemente : boolean : fx   gibt an, ob bei der Ausgabe : "f(x)=" davor stehen soll
Rückgabewert	: String : strFunktion   die auszugebene Funktion

Name	: parsenInString
Funktion	: wandelt die ArrayElemente in einen String um, der die Funktion enthält
Übergabeparameter	: double[]: dArray   enthält die einzelnen Elemente : boolean : fx   gibt an, ob bei der Ausgabe : "f(x)=" davor stehen soll
Rückgabewert	: String : strFunktion   die auszugebene Funktion

Name	: split
Funktion	: sucht in dem Suchstring nach einem Delimiter und splittet den String entsprechend
Übergabeparameter	: String : aSource   enthält den Suchstring : char : aDelimiter   enthält den Delimiter
Rückgabewert	: String[]:   enthält den gesplitteten String in einem Array

## 4.5 Funktionskomplex/Klasse 4: PqFormel

Erstellt für die angegebene Funktion wenn möglich die PQ-Formel und berechnet die Ergebnisse

PqFormel
+dP : Double
+dQ : Double
+strP : String
+strQ : String
+dWurzel : Double
+strFehler : String
+String strEingabeArray : String
+dEingabeArray : Double
+setGrundWerte(ein/aus strArray : String, ein/aus dArray : Double)
+getFunction() : String
+getXWerte() : Double

### 4.5.1 Zustände

static double dP

--> enthält den Wert für "p" als double

static double dQ

--> enthält den Wert für "q" als double

static String strP

--> enthält den Wert für "p" als String

static String strQ

--> enthält den Wert für "q" als String

static double dWurzel

--> enthält das Ergebnis der "Wurzel"

static String strFehler

--> enthält die Fehlermeldung

static String strEingabeArray[]

--> enthält eine mathematische Funktion

static double dEingabeArray[]

--> enthält eine mathematische Funktion

#### 4.5.2 Methoden

Name	: setGrundWerte
Funktion	: Setz die Grundwerte der Klasse fest, die zur Berechnung benötigt werden
Übergabeparameter	: String[]: strArray[] das Eingabearray, das die Funktion enthält : double[]: dArray[] das Eingabearray, das die Funktion enthält
Rückgabewert	: -

Name	: getFunktion
Funktion	: Erstellt die pq-Formel, wobei für "p" und "q" die entsprechenden Werte eingetragen sind
Übergabeparameter	: -
Rückgabewert	: String: strFunction enthält die pq-Formel

Name	: getXWerte
Funktion	: Berechnet die Werte x1 und x2, die bei der pq-Formel als Ergebnis rauskommen
Übergabeparameter	: -
Rückgabewert	: double[]: aWerte[] enthält x1 und x2

## 4.6 Funktionskomplex/Klasse 5: Polinomdivision

Führt für die angegebene mathematische Funktion und einen angegebenen Divisor eine Kurvendiskussion durch

Polinomdivision
+strFehler : String
+strErklaerung : String
+getFunction(ein/aus dArrayEingabe : Double, ein/aus dArrayDif : Double, ein/aus dErgebnis : Double) : Double

### 4.6.1 Zustände

static String strFehler

--> enthält eine Fehlermeldung

static String strErklaerung

--> enthält die Erklärung (die Zwischenschritte)

### 4.6.2 Methoden

Name	: getFunction
Funktion	: Berechnung der Polinomdivision
Übergabeparameter	: double[]: dArrayEingabe   enthält die Ausgangsfunktion
	: double[]: dArrayDif   enthält den Divisor
Rückgabewert	: double[]: dErgebnis

## 4.7 Funktionskomplex/Klasse 6: Ableitung

Berechnet die Ableitung für die angegebene mathematische Funktion

Ableitung
+strFehler : String
+strErklaerung : String
+getAbleitung(ein/aus iGrad : Integer, ein/aus dEingabeWerte : Double) : Double

### 4.7.1 Zustände

static String strFehler

--> enthält eine Fehlermeldung

static String strErklaerung

--> enthält die Erklärung der Zwischenschritte

### 4.7.2 Methoden

Name	: getAbleitung
Funktion	: Ableiten mathematischer Funktionen für die Integralrechnung
Übergabeparameter	: int : iGrad gewünschter Grad der Ableitung : double[]: dEingabeWerte[]   geparsete Werte der Funktion
Rückgabewert	: double[]: dAusgabeWerte[]   enthält die abgeleiteten Elemente

## 4.8 Funktionskomplex/Klasse 7: Nullstellen

Berechnet für die angegebene mathematische Funktion alle Nullstellen

Nullstellen
+strFehler : String
+strErklaerung : String
+getNullstellen(ein/aus dArray : Double) : Double
+getNullstelle(ein/aus dFunktion : Double) : Double
+berechneExponent(ein/aus wert : Double, ein/aus exponent : Integer) : Double
+berechnePolynomDivision(ein/aus dArray : Double, ein/aus dNullstelle : Double) : Double
+berechneX(ein/aus dArray : Double) : Double
+arrayVergroessern(ein/aus array : String) : String

### 4.8.1 Zustände

static String strFehler

--> enthält eine Fehlermeldung

static String strErklaerung

--> enthält die Erklärung für die Zwischenschritte

### 4.8.2 Methoden

Name	: getNullstellen
Funktion	: Berechnet die Nullstellen
Übergabeparameter	: double[]: dArray   enthält die mathematische Funktion
Rückgabewert	: double[]: getNullstellen   enthält alle Nullstellen

Name	: getNullstelle
Funktion	: "rät" eine Nullstelle im Bereich von -10 bis +10
Übergabeparameter	: double[]: dFunktion   enthält die mathematische Funktion
Rückgabewert	: double: getNullstelle   enthält die geratene Nullstelle

Name	: berechneExponent
Funktion	: berechnet den Wert für den Exponenten (Bsp: 3^3)
Übergabeparameter	: double: wert   enthält die Basis des Exponenten : int: exponent   enthält den Exponenten
Rückgabewert	: double: berechneExponent   enthält das Ergebnis

Name	: berechnePolinomDivision
Funktion	: führt die Polinomdivision mit einer Nullstelle aus
Übergabeparameter	: double[]: dArray   enthält die mathem. Funktion : double: dNullstelle   enthält eine Nullstelle
Rückgabewert	: double[]: berechnePolinomDivision   enthält das Ergebnis der Polinomdivision

Name	: berechneX
Funktion	: setzt den X-Wert in eine Funktion ein und berechnet den Y-Wert
Übergabeparameter	: double[]: dArray   enthält die mathem. Funktion
Rückgabewert	: double: berechneX   enthält den Y-Wert

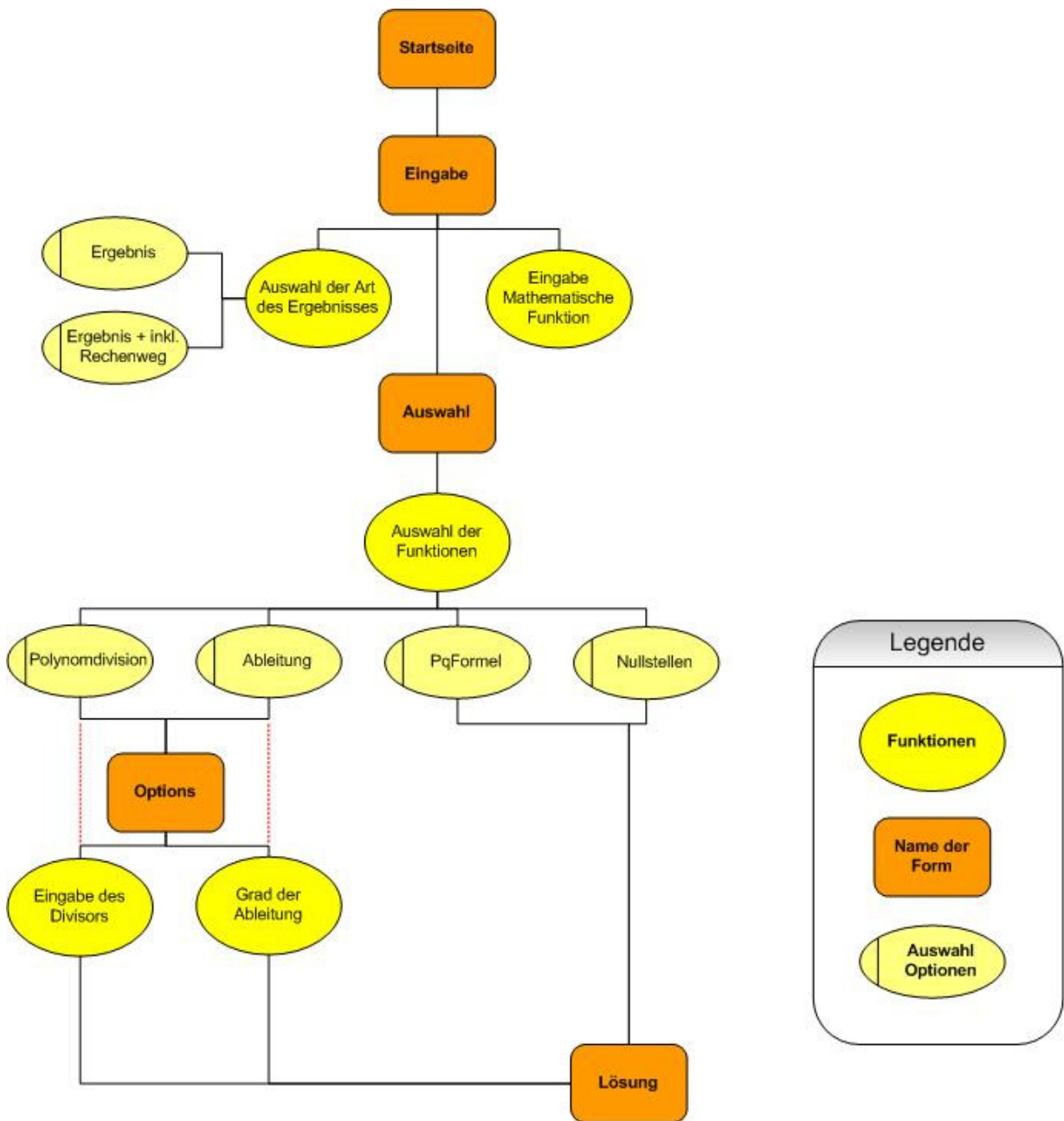
Name	: arrayVergroessern
Funktion	: Vergrößert ein Array um 1 Element
Übergabeparameter	: String[]: array   das zu vergroessernde Array
Rückgabewert	: String[]: tmp   das vergroeußerte Array

# 5 Benutzerschnittstelle

## 5.1 Konzept

Nachdem der User das Programm gestartet hat, wird er über ein Menü durch das Programm geführt.

Der Ablaufplan sieht wie folgt aus:



Zunächst gelangt der User auf die Startseite.

Hier erhält er einige Informationen über KDisk



Über den "weiter" Button geht es auf die EingabeForm:

Hier wird einerseits die mathematische Funktion eingegeben und andererseits festgelegt, wie das Ergebnis dargestellt werden soll.

The screenshot shows the input form of the K-Disk application. The title bar reads "K-DISK". Below the title bar, the text "Bitte Funktion eingeben:" is displayed above a text input field containing the mathematical expression  $1x^2+4x-8$ . Below the input field, the text "Art des Ergebnisses" is displayed above two radio button options: "Nur Ergebnisse der Kurvendiskussion" (which is selected) and "incl Formeln der Kurvendiskussion". At the bottom of the form, there are two buttons: "Exit" on the left and "weiter" on the right.

The screenshot shows the input form of the K-Disk application. The title bar reads "K-DISK". Below the title bar, the text "Bitte Funktion eingeben:" is displayed above a text input field containing the mathematical expression  $1x^2+4x-8$ . Below the input field, the text "Art des Ergebnisses" is displayed above two radio button options: "Nur Ergebnisse der Kurvendiskussion" and "incl Formeln der Kurvendiskussion" (which is selected). At the bottom of the form, there are two buttons: "Exit" on the left and "weiter" on the right.

### a. Vorgaben für die Eingabe der mathematischen Funktion

- von dem "x" muss immer ein Wert stehen

**Korrekte Eingabe:**  $1x^4$

**Inkorrekte Eingabe:**  $x^4$

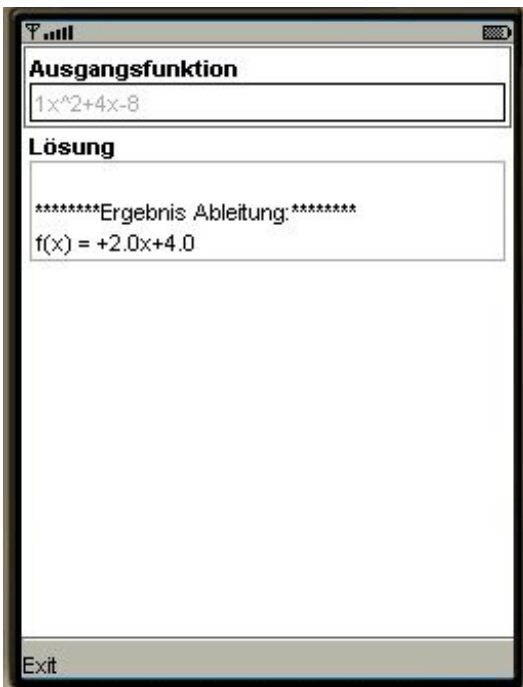
- die Exponenten müssen in absteigender Reihenfolge eingegeben werden
- jeder Exponent darf nur einmal vorhanden sein
- es müssen nicht zwingend ALLE Exponenten absteigend in einer Reihe angegeben werden

**Korrekte Eingabe:**  $2x^4+0x^3+0x^2+2x-0$

**Ebenfalls korrekte Eingabe:**  $2x^4+2x$

- Brüche werden mit einem "/" dargestellt

## b. Art des Ergebnisses

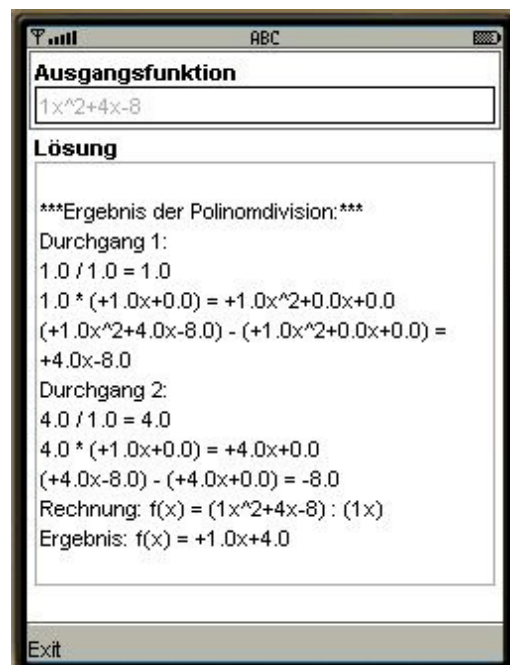


- "Nur Ergebnisse der Kurvendiskussion"

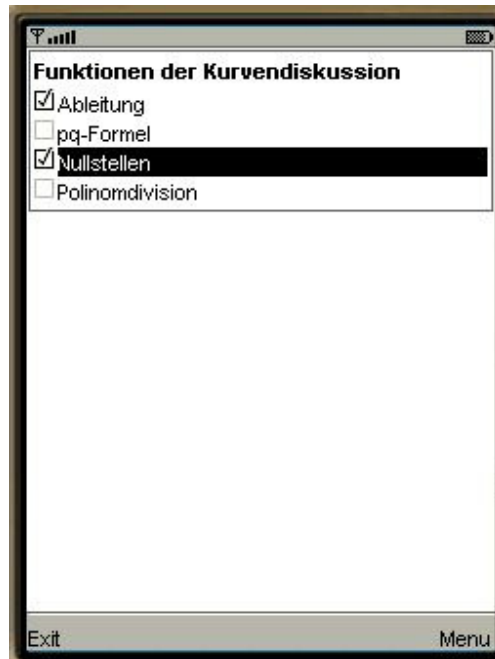
Am Ende des Programms wird nur die Lösung ausgegeben:

- "incl Formeln der Kurvendiskussion"

Am Ende des Programms werden vor der Lösungsausgabe erst alle Rechnungen und Zwischenschritte angezeigt:



## Allgemeine Auswahlmöglichkeiten



### 5.2 Ableitung

Der User wird aufgefordert anzugeben, den wievielten Grad der Ableitung er haben möchte.

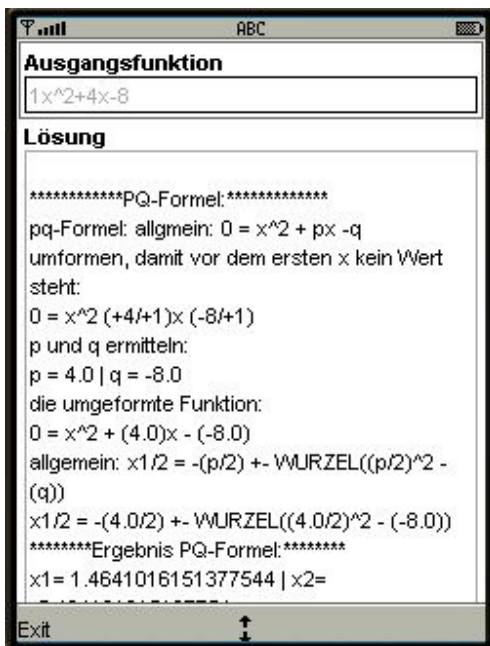


Die Ausgabe für die Ableitung könnte wie folgt aussehen(ohne Zwischenschritte).



### 5.3 PQ-Formel

Die Ausgabe für die PqFormel könnte wie folgt aussehen(mit Zwischenschritte).



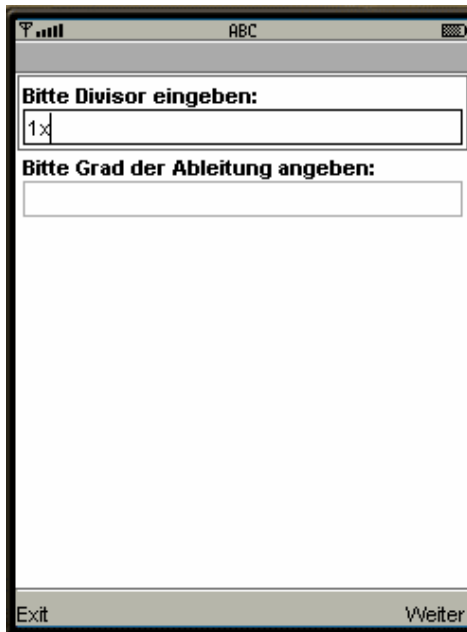
## 5.4 Nullstellen

Die Ausgabe für die Nullstellen könnte wie folgt aussehen(ohne Zwischenschritte).



## 5.5 Polinomdivision

Der User wird aufgefordert den Divisor einzugeben.

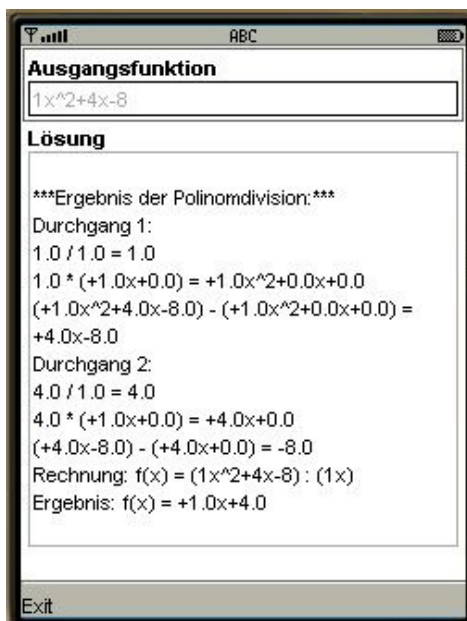


Bitte Divisor eingeben:  
1x

Bitte Grad der Ableitung angeben:

Exit Weiter

Die Ausgabe für die Polinomdivision könnte wie folgt aussehen (mit Zwischenschritte).



Ausgangsfunktion  
1x<sup>2</sup>+4x-8

Lösung

\*\*\*Ergebnis der Polinomdivision:\*\*\*  
Durchgang 1:  
1.0 / 1.0 = 1.0  
1.0 \* (+1.0x+0.0) = +1.0x<sup>2</sup>+0.0x+0.0  
(+1.0x<sup>2</sup>+4.0x-8.0) - (+1.0x<sup>2</sup>+0.0x+0.0) = +4.0x-8.0  
Durchgang 2:  
4.0 / 1.0 = 4.0  
4.0 \* (+1.0x+0.0) = +4.0x+0.0  
(+4.0x-8.0) - (+4.0x+0.0) = -8.0  
Rechnung: f(x) = (1x<sup>2</sup>+4x-8) : (1x)  
Ergebnis: f(x) = +1.0x+4.0

Exit

## 6 **Referenzen**

[1] Java Dok

[2] Netbeans Hilfe

## 7 **Anlagen**

### 7.1 **Dokumentierter Quellcode**

<separate Datei einreichen. Den Programmtext bitte ausdrucken, dabei aber auf *nicht* selbst geschriebene Teile (z.B. verwendete Funktionen der Entwicklungsumgebung, automatisch generierter Programmtext, etc.) verzichten.>

### 7.2 **Eidesstattliche Erklärung**

siehe Anhänge

### 7.3 **Einverständniserklärung der Erziehungsberechtigten**

Alle Teilnehmer unserer Gruppe haben das 18. Lebensjahr vollendet.

### 7.4 **CD-ROM** (Doku und KDisk.jar)