

Übungsaufgabenserie 14
Grundlagen der Informatik und Numerik
Abgabe: 05. 02. 2014, 23:55 Uhr, elektronisch

1. **WurzelBerechner.java**

- (a) Berechnen Sie per Hand mit dem Newtonverfahren $\sqrt[3]{3}$. Beginnen Sie mit dem Näherungswert $x_0 = 3$. Berechnen Sie drei weitere Näherungen. Vergleichen Sie das Ergebnis mit dem mathematisch zu erwartenden Wert (TR). Geben Sie Ihre Berechnung als **.pdf**-Datei ab.
Hinweis: Führen Sie die Rechnung mit gemeinen Brüchen aus.
- (b) Schreiben Sie ein Programm **WurzelBerechner.java**, welches beliebige Wurzeln unter Verwendung der in der Vorlesung eingeführten Klasse `Wurzel` berechnet. Berechnen Sie mit *Ihrem* Programm $\sqrt[3]{3}$, $\sqrt[3]{66.4225}$, $\sqrt[3]{1.96}$, $\sqrt[3]{541.343375}$ und $\sqrt[3]{1.9487171}$. Vergleichen Sie die Ergebnisse mit den Ergebnissen der Java-eigenen Klassenmethode `double Math.pow(double, double)`. Legen Sie die Ausgaben dieser Klasse durch Ausgabenumlenkung in eine Datei **WurzelBerechner.out** ab.
Geben Sie die Datei **WurzelBerechner.out** zusammen mit dem Programm **WurzelBerechner.java** ab.

2. **PolynomNullstellen.java**

Übernehmen Sie aus dem Netz das Programm **PolynomNullstellen.java**, welches Nullstellen für Polynome mit dem Newtonverfahren unter Verwendung der in der Vorlesung eingeführten Klasse `Iteration` berechnet. Verschiedene Nullstellen des Polynoms können mittels dieses Programms durch Änderung der Anfangsnäherung ermittelt werden. Geben Sie Ihre Ergebnisse als **.pdf**-Datei ab:

- (a) Ermitteln Sie für das Polynoms $P_3(x) = x^3 - 5x^2 - 2x + 24$ jeweils eine von den Nullstellen verschiedene Anfangsnäherung, mit welcher die Nullstelle $\bar{x}_1 = 4$, $\bar{x}_2 = 3$ bzw. $\bar{x}_3 = -2$ erreicht wird.
- (b) Wählen Sie ein weiteres Polynom vom Grad 4 mit paarweise verschiedenen Nullstellen als Testbeispiel und berechnen Sie mit dem Programm dessen Nullstellen. Gehen Sie dabei von einer Zerlegung in Linearfaktoren $P_4(x) = (x - \bar{x}_1)(x - \bar{x}_2)(x - \bar{x}_3)(x - \bar{x}_4)$ aus, wobei \bar{x}_i die Nullstellen sind. Erzeugen Sie die normierte Form des Polynoms und versuchen Sie alle Nullstellen durch geeignete, von diesen verschiedenen Anfangsnäherungen zu erreichen.

3. **Arctan.java, ArctanBerechner.java**

Vervollständigen Sie die Klasse `Arctan` (s. hinten) zur Berechnung des Funktionswertes von $\arctan x$:

- (a) Zeichnen Sie zu der Klasse `Arctan` ein Klassendiagramm. Machen Sie kenntlich, wie die neue Klasse mit dem System der Klassen aus der Vorlesung zusammenhängt (Vererbung). Geben Sie das Klassendiagramm in einer **.pdf**-Datei ab.
- (b) Formulieren Sie in der Methode `myArctan` die Berechnung des Funktionswertes von $\arctan x$ als Nullstellenproblem und berechnen Sie diesen unter Verwendung der Newtoniteration und ausschließlich mit den Klassen aus der Vorlesung.
Hinweis: Sollten Sie die Klasse `Tangens` aus der Serie 12 nicht oder fehlerhaft programmiert haben, verwenden Sie die Java-eigene Klassenmethode `double Math.tan(double)`.
- (c) Das Verfahren konvergiert für $|x| > 1$ sehr schlecht. Verbessern Sie in der Methode `myArctanBesser`

Ihre Berechnung, indem Sie für diesen Fall die Beziehung $\arctan x = 2 \arctan \frac{x}{1 + \sqrt{1 + x^2}}$ nutzen. Berechnen

Sie die Wurzel mit der entsprechenden Klasse aus der Vorlesung. Geben Sie die Datei **Arctan.java** ab.

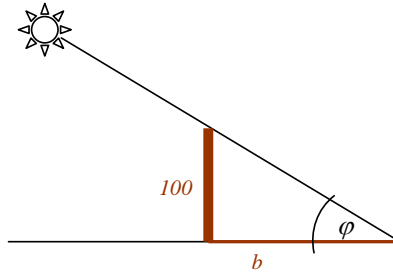
- (d) Berechnen Sie unter Verwendung *Ihrer* Klasse `Arctan` in einem Programm **ArctanBerechner.java** die Funktionswerte $\arctan x$ für die Argumente $x = 0$, $2 - \sqrt{3}$, $\sqrt{1 - \frac{2}{5}\sqrt{5}}$, $\sqrt{2} - 1$, $\frac{1}{3}\sqrt{3}$, $\sqrt{5 - 2\sqrt{5}}$, 1 , $\sqrt{3}$ und

10. Nutzen Sie für die Überprüfung der Korrektheit die Beziehung $x = \tan(\arctan x)$ mit der Klasse `Tangens` als Umkehrfunktion. Schreiben Sie die mit einem Taschenrechner berechneten Eingabewerte in eine Datei **ArctanBerechner.in** und die Ausgabewerte *Ihres* Programms in eine Datei **ArctanBerechner.out**.

Geben Sie die Dateien **ArctanBerechner.in**, **ArctanBerechner.out** zusammen mit dem Programm **ArctanBerechner.java** ab.

4. SonnenStand.java

Aus der Länge b des Schattens, den ein senkrechter, 100 cm langer Stab auf eine horizontale Fläche wirft, lässt sich der Winkel φ bestimmen, welcher die Sonnenstrahlen gegen die Horizontale bildet. Er wird Sonnenhöhe genannt. Man erhält den Winkel φ aus $\tan(\varphi) = \frac{100}{b}$:



- (a) Schreiben Sie ein Anwendungsprogramm *SonnenStand.java*, welches diesen Winkel [GRD] berechnet. Verwenden Sie aus der 3. Aufgabe dieser Serie bei der Umrechnung die Klasse *Arctan*. Sollten Sie die Klasse *Arctan* nicht oder fehlerhaft programmiert haben, verwenden Sie die Java-eigene Klassenmethode `double Math.atan(double)`.
- (b) Berechnen Sie die Sonnenhöhe in [GRD] für die Schattenlängen, die im Verlauf eines Tages gemessen wurden:

Zeit	6:00 Uhr	9:00 Uhr	12:00 Uhr	15:00 Uhr	18:00 Uhr
b	224 cm	1 m	81 cm	119 cm	247 cm

Schreiben Sie die Eingabewerte in eine Datei *SonnenStand.in* und die Ausgabewerte *Ihres* Programm in eine Datei *SonnenStand.out*.
Geben Sie die Dateien *SonnenStand.java*, *SonnenStand.in* und *SonnenStand.out* ab.
Hinweis: Beachten Sie, dass Winkelfunktionen stets mit dem Bogenmaß arbeiten.

Arctan.java (Grobstruktur)

```
// Arctan.java MM 2013

/**
 * Logarithmusfunktion f(x) = arctan( x),
 * D = R, W = ] -PI/2, PI/2[.
 */
public class Arctan extends DifferenzierbareFunktion
{
    /* ----- */
    // service-Methode
    /**
     * Berechnen eines Funktionswertes.
     * @param arg Argument
     * @return arctan( arg)
     */
    public double wert( double arg)
    {
        // Trivialfall
        if( arg == 0) return 0;

        // einfache Newtoniteration
        // return myArctan( arg);
        // verbesserte Newtoniteration
        return myArctanBesser( arg);
    }
}
```

```
/**
 * Berechnen eines Funktionswertes als Nullstelle der
 * Umkehrfunktion  $\tan(x) - \arg$  mittels Newton.
 * @param arg Argument
 * @return arctan( arg)
 */
private double myArctan( double arg)
{
// Newtoniteration (b)

}

/**
 * Verbesserte Funktionswertberechnung fuer  $|x| > 1$ .
 * @param arg Argument
 * @return arctan( arg)
 */
private double myArctanBesser( double arg)
{
// Verbesserung (c)

}

/**
 * Berechnen einer ersten Ableitung
 *  $f'(x) = 1 / (1 + x^2)$ .
 * @param arg Argument
 * @return f'( arg)
 */
public double wertErsteAbleitung( double arg)
{
// Trivialfall
    if( arg == 0) return 0;

// Sonst
    return 1 / ( 1 + arg * arg);
}

/* ----- */
// toString-Methode
/**
 * Darstellen der Arcustangensfunktion.
 * @return Funktion in linearer Schreibweise
 */
public String toString()
{
    return "arctan( x)";
}
}
```