Das HeronVerfahren für root(a)
(Babilonisches Wurzelziehen)

Ein weiteres Annäherungsverfahren für das Wurzelziehen
Ein Spezialfall des Newton Verfahrens

von
Bieker Sebastian
11038605
Gruppe F Grün

Geschichte:

Heron war wohl einer der bedeutensten Wissenschaftler seiner Zeit.
Er war Mathematiker, Mechaniker, Physiker, Naturforscher, Techniker und Ingenieur der Antike.
Man weiß das er in der Zeitspanne 150 v. Chr. - 250 n. Chr. im ägyptichen Alexandrien gelebt hat.
Seine Schriften überlebten in griechisch, lateinisch und arabischen Übersetzungen.

Das HeronVerfahren geht eigentlich schon auf die Babilonier zurück welche schon ca. 2000 Jahre vor Heron gelebt haben. Daher wird das Verfahren heute eher als Babilonisches Wurzelziehen bezeichnet. Der Name HeronVerfahren kommt wohl daher das er als erstes das Verfahren schriftlich niedergelegt hat.

Er beschreibt das Verfahren in seinem Buch Metrica

Bestehend aus 3 Büchern über geometrische Flächen- und Körperberechnungen:

Teil 1 handelt von Flächeninhalten von Dreiecken und anderen Polygonen mit 4 bis 12 Seiten, Oberflächen von Pyramiden, Zylindern, Prismen Kugeln usw.

Teil 2 handelt von der Bestimmung von Volumina von Kugeln, Zylindern, Prismen, Pyramiden u. s. w.

Teil 3 handelt von der Aufteilung von Flächen und Volumina in bestimmten Verhältnissen. Er gibt eine Methode zum Auffinden der dritten Wurzel einer Zahl an und berechnet die dritte Wurzel aus 100.

Das Heron Verfahren ist ein alter iterativer Algorithmus zur Bestimmung einer rationalen Näherung der Quadratwurzel einer Zahl. Es ist ein Spezialfall des Newton Verfahrens.

Die Itereationsvorschrift lautet:

$$x_{n+1} = \frac{x_n + \frac{a}{x_n}}{2}.$$

Hierbei steht *a* für die Zahl, deren Quadratwurzel bestimmt werden soll. Der Startwert x_0 der Iteration kann, solange er nicht gleich Null ist beliebig festgesetzt werden.

Geometrische Veranschaulichung:

Der Flächeninhalt eines Quadrates kann über das Quadrat der Länge seiner Seiten bestimmt werden. Die Bestimmung der Quadratwurzel kann also so gedeutet werden als Bestimmung der Seitenlänge eines Quadrates mit dem Flächeninhalt a.

Die Idee hinter dem Heron Verfahren ist nun, von dem Flächeninhalt eines Rechecks auszugehen, und die Seitenlängen so anzunähern das ein Quadrat entsteht.

Hierzu wird eine Startwert gewählt.

Wenn die Quadrat Wurzel aus a=9 gezogen werden soll, nehmen wir z.B. den Startwert x_0=1 an.

Dies bedeutet Geometrisch, das das Rechteck nun eine Seitenlänge von $x_0 = 1$ besitzt.

Die andere Seitenlänge dieses Rechtecks ergibt sich aus dem vorgegebenen Flächeninhalt:

$$y_0 = \frac{9}{1} = 9$$

Bei der Betrachtung ist unmittelbar ersichtlich, dass es eine geeignetere Näherung an ein Quadrat gibt, denn die eine Seitenlänge $x_0 = 1$ ist zu klein, die andere mit $y_0 = 9$ zu groß.

Eine bessere Annäherung an die Quadratseiten kann nun über das arithmetische Mittel der Seitenlängen $x_0 = 1$ und $y_0 = 9$ erfolgen. (Es gibt eine Ganze Reihe von weiteren Möglichkeiten um das Verfahre zun verfeinern)

$$x_1 = \frac{x_0 + y_0}{2} = \frac{9+1}{2} = 5$$

Die Länge der zweiten Seite ergibt sich wieder durch den Flächeninhalt a.

$$y_1 = \frac{a}{x_1} = \frac{9}{5} = 1{,}8$$

Die Werte $x_1 = 5$ und $y_1 = 1{,}8$ sind geometrisch gedeutet die Seitenlängen eines zweiten Näherungs-Rechtecks. Dieses und die folgenden Rechtecke lassen sich nun weiter verbessern durch erneute Bildung des arithmetischen Mittelwertes als verbesserte Näherung an die Seitenlänge eines Quadrates mit der Seitenlänge $\sqrt{a}$.

Anwendung:

Die Umsetzung im Java Code sieht folgender maßen aus.

```
public class HeronSqrt {
	public static BigDecimal[] heronSqrt(double x, double a, int scale, int count){
		BigDecimal[] bx = new BigDecimal[count+1];
		bx[0] = new BigDecimal("" + x).setScale(scale);
		BigDecimal ba = new BigDecimal("" + a).setScale(scale);
		if(x==0)
			System.out.println("x darf nicht null sein bitte geben eine andere Zahl ein");
		else{
			for(int i=0;i<count;i++){
				bx[i+1] =
	bx[i].add(ba.divide(bx[i],BigDecimal.ROUND_HALF_DOWN)).divide(new
	BigDecimal(""+2)).setScale(scale,BigDecimal.ROUND_HALF_DOWN);
				System.out.println((i+1)+". Druchlauf: " + bx[i+1]);
			}
		}
		return bx;
	}

	public static BigDecimal[] heronSqrtY(BigDecimal[] bx, double a, int scale, int count){
		BigDecimal[] by = new BigDecimal[count+1];
		BigDecimal ba = new BigDecimal("" + a).setScale(scale);
		for(int i=0;i<=count;i++){
			by[i]=ba.divide(bx[i],BigDecimal.ROUND_HALF_DOWN).setScale(scale);
		}
		return by;
	}

	public static void main(String[] args){
	}
}
```

Implementierung einer graphischen Lösung des Verfahrens:

Um anschaulich darstellen zu können wie das Verfahren funktioniert, entschloss ich mich zu einer GUI welche die Eingabe mehrere Variabeln zuließ.

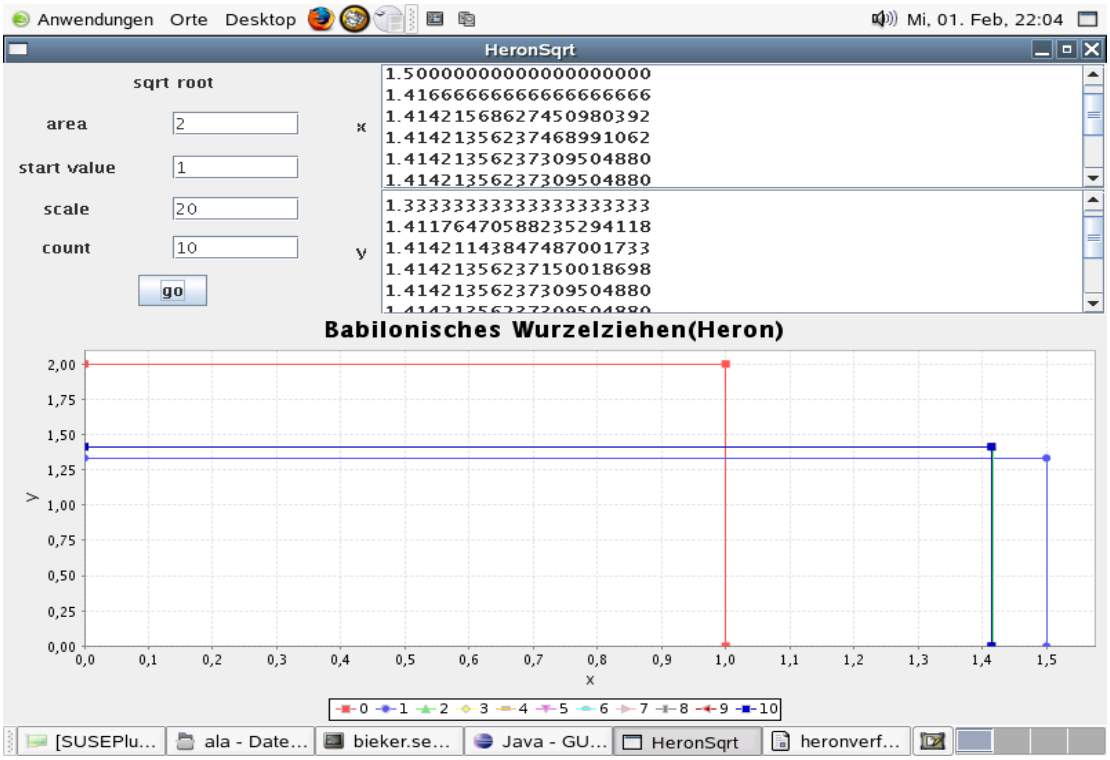
Als erstes muss die Zahl angegeben werden aus der die Wurzel gezogen werden soll.

Im Feld start value wird der Wert angegeben von dem aus sich an die Wurzel angenähert werden soll.

Um so genauer dieser Wert geschätzt wird desto weniger Schritte benötigt die Iteration um den exakten Wurzelwert zu finden.

Das Feld scale gibt die Anzahl der Nachkommastellen an auf die die Wurzel genau berechnet werden soll.

Und im Feld count kann geschätzt werden wie viele Schritte der Algorithmus benötigt um die exakte Wurzel zu finden.



Als Ergebnis sieht man nun die Werte der einzelnen Iterationsschritte.

In der Liste ist zu erkennen das sich die Werte an einander annähern.

Der x Wert nähert sich von Oben und der y Wert nähert sich von Unten der eigentlichen Wurzel.

In dem Koordinatensystem ist nocheinmal anschaulich darstellt wie sich die einzelnen Seiten annähern.

Fazit:

Das Babilonische Wurzelziehen ist ein einfaches aber sehr einfallsreiches und effektives Verfahren um eine Quadratwurzel beliebig genau zu bestimmen.