Lektion 5

do-while-Schleife

Modellierung

Quadratwurzelberechnung

Pi-Berechnung

do-while-Schleife

Wir wollen ein Ratespiel entwickeln!

Der Computer denkt sich eine Zahl zwischen 1 und 100 aus.

Der Spieler rät, bis er die Zahl erraten hat, und erhält nach jedem Versuch einen Hinweis, ob die geratene Zahl größer oder kleiner ist.

Der Computer denkt sich eine Zahl zwischen 1 und 100 aus.

Math.random
liefert eine Fließkommazahl
x mit 0 <= x < 1

```
int zufallszahl = (int) (Math.random() * 100 + 1);
```

Der Spieler rät bis er die Zahl erraten hat…

Schleife mit Eingabe

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);

WIEDERHOLE BIS ERRATEN
  int eingabe = scanner.nextInt();
```

… und erhält nach jedem Versuch einen Hinweis, ob die geratene Zahl größer oder kleiner ist.

if-Abfrage Ausgabe

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);

WIEDERHOLE BIS ERRATEN
  int eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
```

Jetzt benötigen wir noch die Schleife…

do-while-Schleife

- Eine `do-while`-Schleife wiederholt den Schleifenrumpf, solange die Laufbedingung erfüllt ist.
- Die Laufbedingung wird am Ende der Schleife überprüft (**fußgesteuert**).
- Der Schleifenrumpf wird daher **mindestens einmal** ausgeführt.
- Die `do-while`-Schleife hat folgende Form:

do-while-Schleife

- Eine `do-while`-Schleife wiederholt den Schleifenrumpf, solange die Laufbedingung erfüllt ist.
- Die Laufbedingung wird am Ende der Schleife überprüft (**fußgesteuert**).
- Der Schleifenrumpf wird daher **mindestens einmal** ausgeführt.
- Die `do-while`-Schleife hat folgende Form:

```
do
{
  <Anweisungen>
}
while(<bool‘scher Ausdruck>);
```

Schleifenrumpf

Laufbedingung

do-while-Schleife - Beispiel

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);

do
{
  int eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (NOCH NICHT ERRATEN);
```

do-while-Schleife - Beispiel

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);

do
{
  int eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe);
```

do-while-Schleife - Beispiel

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);

do
{
  int eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe);
```

eingabe ist sichtbar

eingabe ist nicht sichtbar

do-while-Schleife - Beispiel

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;

do
{
  eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe);
```

do-while-Schleife - Beispiel

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;

do
{
  eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe);
```

eingabe ist sichtbar

break

```
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;

do
{
  eingabe = scanner.nextInt();
  if (zufallszahl == eingabe)
  {
    System.out.println("Volltreffer!");
    break;
  }
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (true);
```

mit break lässt sich eine Schleife vorzeitig verlassen

Die Ausführung wird dahinter fortgesetzt.

Die Laufbedingung wird dadurch einfacher.
Der Programmfluss kann - vor allem bei häufiger Verwendung - unübersichtlicher werden.

Wir erweitern unser Spiel dahingehend,
dass der Spieler nur 10 Rateversuche hat.

do-while-Schleife - Beispiel

```
int anzahlVersuche = 0;
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;
do
{
  eingabe = scanner.nextInt();
  anzahlVersuche++;
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe && anzahlVersuche < 10);
if (zufallszahl == eingabe) System.out.println("Sie haben gewonnen.");
else System.out.println("Sie haben verloren.");
```

do-while-Schleife - Beispiel

```
int anzahlVersuche = 0;
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;
do
{
  eingabe = scanner.nextInt();
  anzahlVersuche++;
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe && anzahlVersuche < 10);
if (zufallszahl == eingabe) System.out.println("Sie haben gewonnen.");
else System.out.println("Sie haben verloren.");
```

Was können wir noch vereinfachen?

Hinweis: DRY

do-while-Schleife - Beispiel

```
int anzahlVersuche = 0;
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;
do
{
  eingabe = scanner.nextInt();
  anzahlVersuche++;
  if (zufallszahl == eingabe)
    System.out.println("Volltreffer!");
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (zufallszahl != eingabe && anzahlVersuche < 10);
if (zufallszahl == eingabe) System.out.println("Sie haben gewonnen.");
else System.out.println("Sie haben verloren.");
```

Doppelter Code

```java
boolean getroffen = false;
int anzahlVersuche = 0;
int zufallszahl = (int) (Math.random() * 100 + 1);
Scanner scanner = new Scanner(System.in);
int eingabe;
do
{
  eingabe = scanner.nextInt();
  anzahlVersuche++;
  if (zufallszahl == eingabe)
  {
    getroffen = true;
    System.out.println("Volltreffer!");
  }
  else if (zufallszahl > eingabe)
    System.out.println("Die gesuchte Zahl ist größer!");
  else if (zufallszahl < eingabe)
    System.out.println("Die gesuchte Zahl ist kleiner!");
}
while (!getroffen && anzahlVersuche < 10);
if (getroffen) System.out.println("Sie haben gewonnen.");
else System.out.println("Sie haben verloren.");
```

Wir arbeiten mit einer sogenannten **Flag** zur Markierung.

Wir wollen programmatisch herausfinden, gegen welchen Wert die folgende Reihe voraussichtlich konvergiert.

$$\sum_{k=0}^{\infty} \frac{1}{3^k}$$

Dazu schreiben wir die ersten Glieder auf:

$$\sum_{k=0}^{\infty} \frac{1}{3^k} = \frac{1}{3^0} + \frac{1}{3^1} + \frac{1}{3^2} + \frac{1}{3^3} + \frac{1}{3^4} + \cdots$$

$$\sum_{k=0}^{\infty} \frac{1}{3^k} = \frac{1}{3^0} + \frac{1}{3^1} + \frac{1}{3^2} + \frac{1}{3^3} + \frac{1}{3^4} + \cdots$$

Wir müssen die Glieder nacheinander aufsummieren.

➢ wir brauchen eine Schleife

Der Exponent im Nenner
bzw. der Nenner
bzw. der Summand
ändert sich in jedem Durchlauf.

$$\sum_{k=0}^{\infty} \frac{1}{3^k} = \frac{1}{3^0} + \frac{1}{3^1} + \frac{1}{3^2} + \frac{1}{3^3} + \frac{1}{3^4} + \cdots$$

Berechnung des ersten Summanden:

```
summand = 1.0/Math.pow(3, 0);
```

Berechnung des zweiten Summanden:

```
summand = 1.0/Math.pow(3, 1);
```

Berechnung des dritten Summanden:

```
summand = 1.0/Math.pow(3, 2);
```

Berechnung des (k+1)-ten Summanden:

```
summand = 1.0/Math.pow(3, k);
```

$$\overbrace{a + b}^{\text{Summe}}$$

Summand Summand

```
double summe = 0;
double summand;


WIEDERHOLE
   summand = 1.0/Math.pow(3, k);
   summe = summe + summand;
```

k muss in jedem
Durchlauf erhöht werden

do-while-Schleife, weil mindestens
eine Berechnung stattfindet

Welche Laufbedingung kommt in Frage?

- unendlich?
- bis k oder 3^k an die Zahlengrenze stößt?
- bis die Wertänderung unter ein bestimmtes EPSILON fällt?

```
double summe = 0;
double summand;
int k = 0;
do
{
  summand = 1.0/Math.pow(3, k);
  summe = summe + summand;
  k++;
}
while(                    );
System.out.println(summe);
```

```java
double summe = 0;
double summand;
int k = 0;
do
{
  summand = 1.0/Math.pow(3, k);
  summe = summe + summand;
  k++;
}
while(summand > 1E-10);
System.out.println(summe);
```

wenn weniger als 0.0000000001 zur Gesamtsumme dazukommt: Abbruch

Äquivalenz do-while und while

- **Sei b eine bool‘sche Variable, die einen bool‘schen Ausdruck gespeichert habe.**

```
while(b) {
  <Anweisungen>
}
```

```
if(b) {
  do {
    <Anweisungen>
  }
  while(b);
}
```

```
do {
  <Anweisungen>
}
while(b);
```

```
<Anweisungen>
while(b)
{
  <Anweisungen>
}
```

continue

- Mit der Anweisung **continue** ist es möglich, den aktuellen Schleifendurchlauf abzubrechen und mit dem nächsten zu beginnen.

```
/*Addiere alle Zahlen von 1 bis 10, lasse aber dabei alle durch 5
teilbaren Zahlen aus*/
int j = 0;
int sum = 0;
while(j < 10)
{
  j++;
  if (j % 5 == 0) continue;
  sum = sum + j;
}
System.out.println(sum);
```

Endlosschleifen

- Folgende Schleifen laufen endlos, es sei denn sie werden durch **break** oder durch eine Fehlermeldung (Exception, Error) verlassen:

```
for (;;)
{
  <Anweisungen>
}
```

⟺

```
while(true)
{
 <Anweisungen>
}
```

⟺

```
do
{
 <Anweisungen>
}
while(true);
```

Modellierung von Kontrollstrukturen

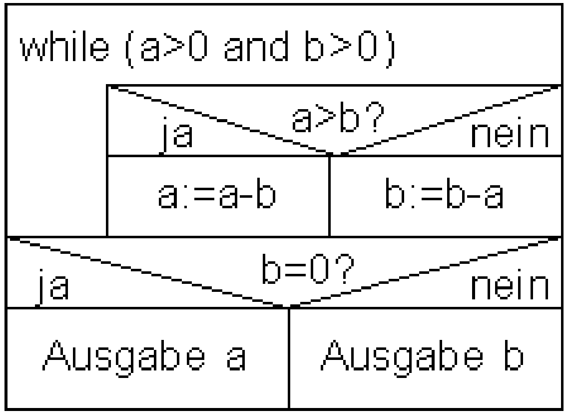
Modellierung von Kontrollstrukturen

- Mit Kontrollstrukturen (Schleifen, Verzweigungen) kann der Programmfluss gesteuert werden.
- Zur Modellierung des Programmflusses gibt es verschiedene Möglichkeiten:
  - Pseudocode

```
WIEDERHOLE bis eingabe == -1
  summe = summe + eingabe;
```

Modellierung von Kontrollstrukturen

- Mit Kontrollstrukturen (Schleifen, Verzweigungen) kann der Programmfluss gesteuert werden.
- Zur Modellierung des Programmflusses gibt es verschiedene Möglichkeiten:
  - Struktogramme/Nassi-Shneiderman-Diagramme

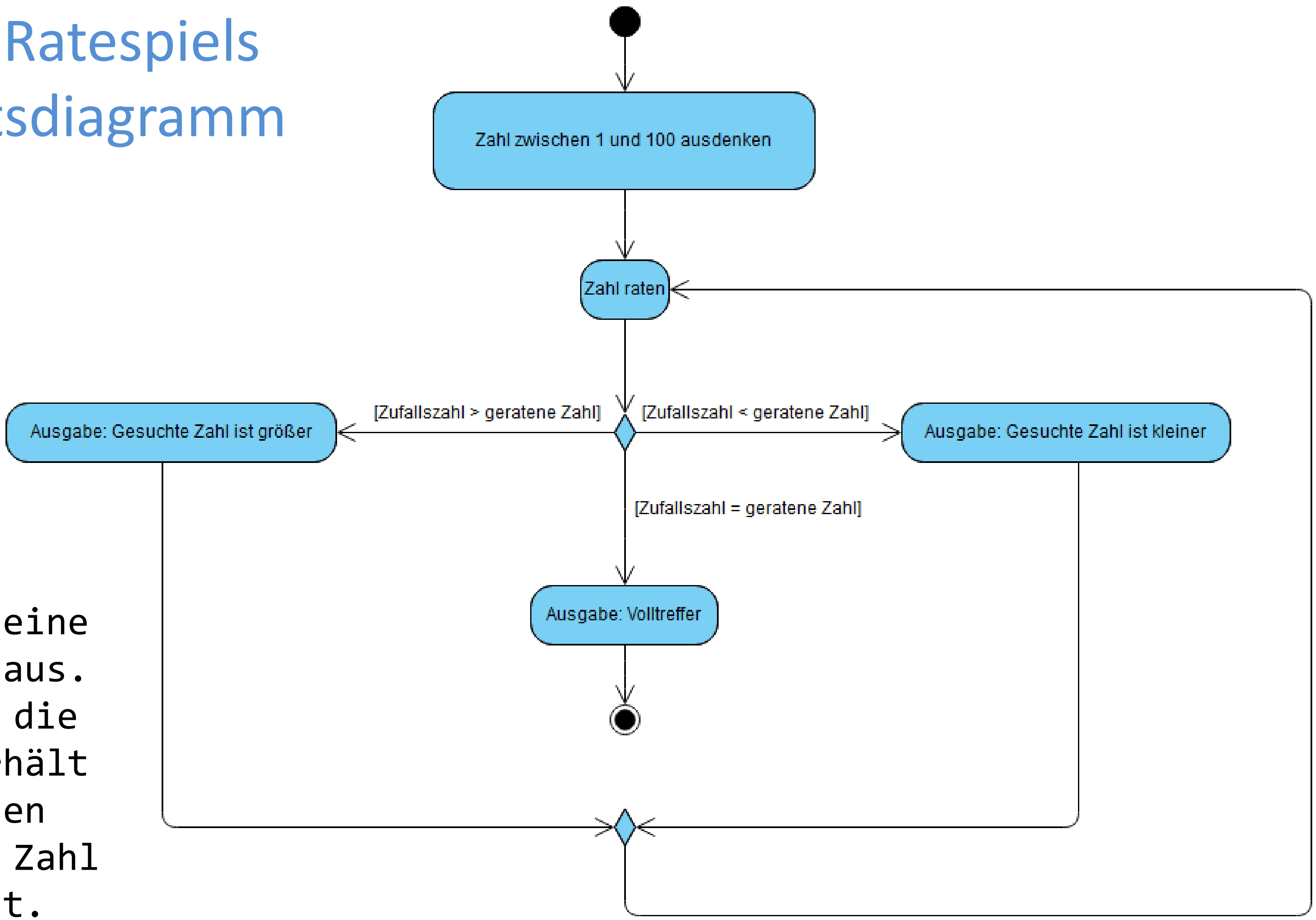


source: http://upload.wikimedia.org/wikipedia/commons/3/33/NassiShneiderman.png

Modellierung von Kontrollstrukturen

- Mit Kontrollstrukturen (Schleifen, Verzweigungen) kann der Programmfluss gesteuert werden.
- Zur Modellierung des Programmflusses gibt es verschiedene Möglichkeiten:
  - Flussdiagramme/Programmablaufpläne
  - Aktivitätsdiagramme
  - …

Modellierung des Ratespiels mit einem Aktivitätsdiagramm



Der Computer denkt sich eine Zahl zwischen 1 und 100 aus. Der Spieler rät, bis er die Zahl erraten hat, und erhält nach jedem Versuch einen Hinweis, ob die geratene Zahl größer oder kleiner ist.

Quadratwurzelbestimmung

Quadratwurzel bestimmen

$$\sqrt{x} = y_{ges}, \text{ wobei } x, y_{ges} \in \mathbb{R}^+ \backslash \{0\}$$

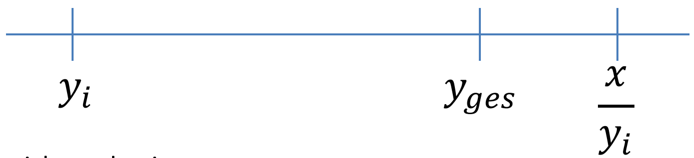
$$\Leftrightarrow \quad x = y_{ges} \cdot y_{ges}, \text{ da } x, y_{ges} > 0$$

$$\Leftrightarrow \frac{x}{y_{ges}} = y_{ges}$$

Setzt man nun einen Näherungswert y_i für y_{ges} mit $y_i < y_{ges}$ ein, gilt:

Aus $y_i < y_{ges} \Rightarrow \frac{x}{y_i} = \frac{y_{ges} \cdot y_{ges}}{y_i} = \underbrace{\frac{y_{ges}}{y_i}}_{>1} y_{ges} > y_{ges}$

D. h. $y_i < y_{ges} \Rightarrow \frac{x}{y_i} > y_{ges}$



y_i bezeichnet den i-ten Näherungswert

Quadratwurzel bestimmen

$$\sqrt{x} = y_{ges}, \text{ wobei } x, y_{ges} \in \mathbb{R}^+ \backslash \{0\}$$

$$\Leftrightarrow \quad x = y_{ges} \cdot y_{ges}, \text{ da } x, y_{ges} > 0$$

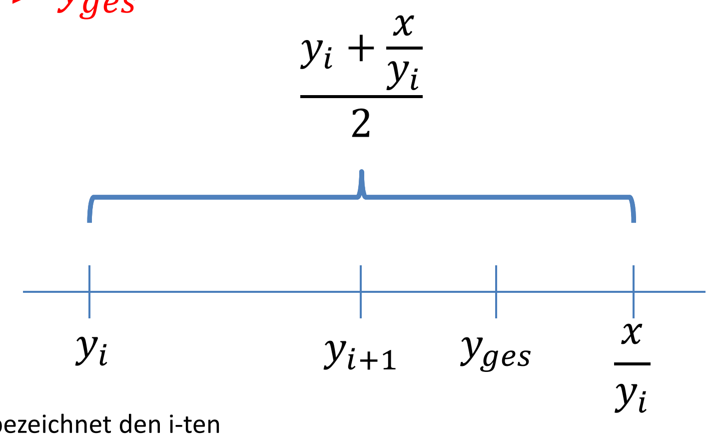
$$\Leftrightarrow \frac{x}{y_{ges}} = y_{ges}$$

Setzt man nun einen Näherungswert y_i für y_{ges} mit $y_i < y_{ges}$ ein, gilt:

Aus $y_i < y_{ges} \Rightarrow \frac{x}{y_i} = \frac{y_{ges} \cdot y_{ges}}{y_i} = \underbrace{\frac{y_{ges}}{y_i}}_{>1} y_{ges} > y_{ges}$

D. h. $y_i < y_{ges} \Rightarrow \frac{x}{y_i} > y_{ges}$

$\Rightarrow \boxed{y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}}$ liegt näher an y_{ges}



y_i bezeichnet den i-ten Näherungswert

Quadratwurzel bestimmen

$$\sqrt{x} = y_{ges}, \text{ wobei } x, y_{ges} \in \mathbb{R}^{+}\backslash\{0\}$$

$$\Leftrightarrow \quad x = y_{ges} \cdot y_{ges}, \text{ da } x, y_{ges} > 0$$

$$\Leftrightarrow \frac{x}{y_{ges}} = y_{ges}$$

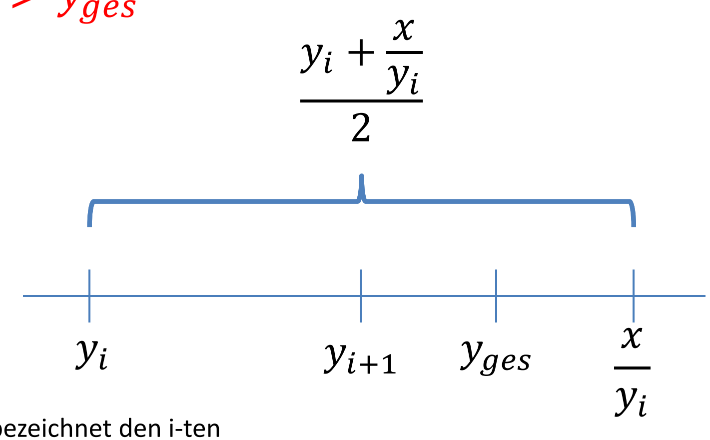
Setzt man nun einen Näherungswert y_i für y_{ges} mit $y_i < y_{ges}$ ein, gilt:

Aus $y_i < y_{ges} \Rightarrow \frac{x}{y_i} = \frac{y_{ges} \cdot y_{ges}}{y_i} = \underbrace{\frac{y_{ges}}{y_i}}_{>1} y_{ges} > y_{ges}$

D. h. $y_i < y_{ges} \Rightarrow \frac{x}{y_i} > y_{ges}$

$\Rightarrow \boxed{y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}}$ liegt näher an y_{ges}

Analog gilt: $y_i > y_{ges} \Rightarrow \frac{x}{y_i} < y_{ges}$



y_i bezeichnet den i-ten Näherungswert

Quadratwurzel bestimmen

- Wann breche ich die Berechnung ab?
- Bei einer bestimmten Genauigkeit, z. B.

$$\left|\frac{x}{y_i} - y_i\right| < 10^{-10}$$

Zum Nachlesen: Quadratwurzel (I)

Sei $y_{ges} \in \mathbb{R}^+$ das gesuchte Ergebnis der Wurzel und $y_{ges} \neq 0$. Sei $x \in \mathbb{R}^+$ der Wert, aus dem die Wurzel gezogen wird. Sei $y_i \in \mathbb{R}^+$ für $i \in \mathbb{N}$ der i-te Näherungswert für y_{ges}.

$$\sqrt{x} = y_{ges}$$

$$x = y_{ges} \cdot y_{ges}$$

$$\frac{x}{y_{ges}} = y_{ges}$$

$$\frac{x}{y_{ges}} - y_{ges} = 0$$

Setzt man nun einen Näherungswert y_i für y_{ges} ein, gilt:

$$y_i < y_{ges} \Rightarrow \frac{x}{y_i} = \frac{y_{ges} \cdot y_{ges}}{y_i} > y_{ges}$$

$$y_i > y_{ges} \Rightarrow \frac{x}{y_i} = \frac{y_{ges} \cdot y_{ges}}{y_i} < y_{ges}$$

Zum Nachlesen: Quadratwurzel (II)

Daher erhält man einen Wert der kleiner und einen der größer als der gesuchte Wert ist. Der wirkliche Wert liegt zwischen den beiden Werten. Daher bildet man die Mitte zwischen den beiden Werten, um einen neuen Näherungswert zu erhalten:

$$y_{i+1} = \frac{\frac{x}{y_i} + y_i}{2}$$

Den neuen Näherungswert kann man wieder für y_{ges} einsetzen. Da man eine Lösung nicht immer genau bestimmen (siehe z.B. $\sqrt{2}$), wird dieser Schritt solange wiederholt, bis eine bestimmte Genauigkeit erfüllt ist, z.B.:

$$\left| \frac{x}{y_i} - y_i \right| < 0.000000000000001 = 10^{-15}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\;$ wähle $y_0 = 1$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\;$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1,5$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1{,}5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\ \text{wähle } y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1,5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2} = \frac{\frac{2}{1,5} + 1,5}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1,5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2} = \frac{\frac{2}{1,5} + 1,5}{2} = \frac{\frac{4}{3} + \frac{3}{2}}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\ \text{wähle } y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1{,}5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2} = \frac{\frac{2}{1{,}5} + 1{,}5}{2} = \frac{\frac{4}{3} + \frac{3}{2}}{2} = \frac{\frac{17}{6}}{2}$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\;$ wähle $y_0 = 1$

$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1,5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2} = \frac{\frac{2}{1,5} + 1,5}{2} = \frac{\frac{4}{3} + \frac{3}{2}}{2} = \frac{\frac{17}{6}}{2} = \frac{17}{12} =$$

Quadratwurzel: Beispiel

$$y_{i+1} = \frac{y_i + \frac{x}{y_i}}{2}$$

Näherungweise Berechnung von $\sqrt{2}$

$x = 2,\ \text{wähle } y_0 = 1$

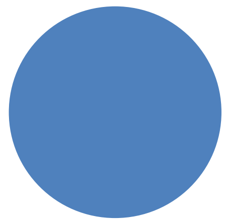
$$y_1 = \frac{\frac{x}{y_0} + y_0}{2} = \frac{\frac{2}{1} + 1}{2} = \frac{3}{2} = 1{,}5$$

$$y_2 = \frac{\frac{x}{y_1} + y_1}{2} = \frac{\frac{2}{1{,}5} + 1{,}5}{2} = \frac{\frac{4}{3} + \frac{3}{2}}{2} = \frac{\frac{17}{6}}{2} = \frac{17}{12} = 1{,}41\overline{6}$$

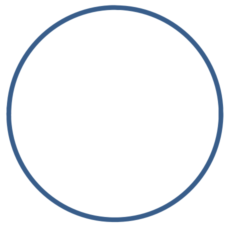
PI-Berechnung

π-Beispiel:

Kreisfläche: πr^2



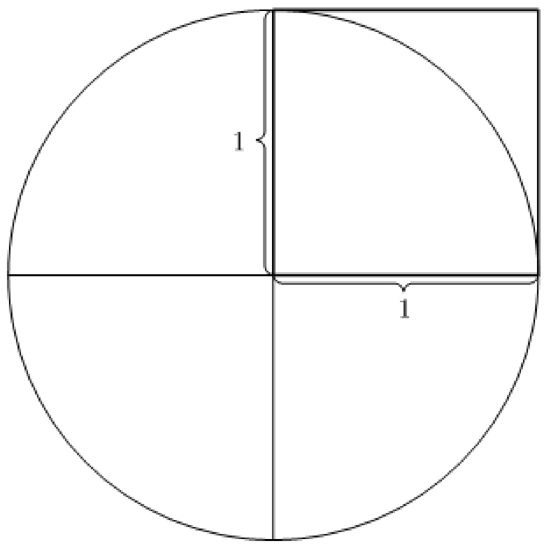
Kreisumfang: $2\pi r$



Aber wie kommt man auf den Wert für π?

Kreisfläche Rechtecknäherung (I)

Betrachten wir für eine erste Näherung
den Einheitskreis mit r = 1.

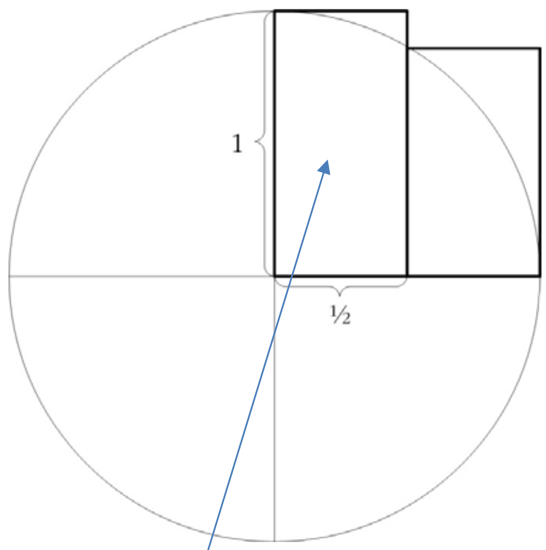


$$\frac{\pi}{4} \approx 1 \cdot 1$$

$$\pi \approx 4$$

Kreisfläche Rechtecknäherung (II)

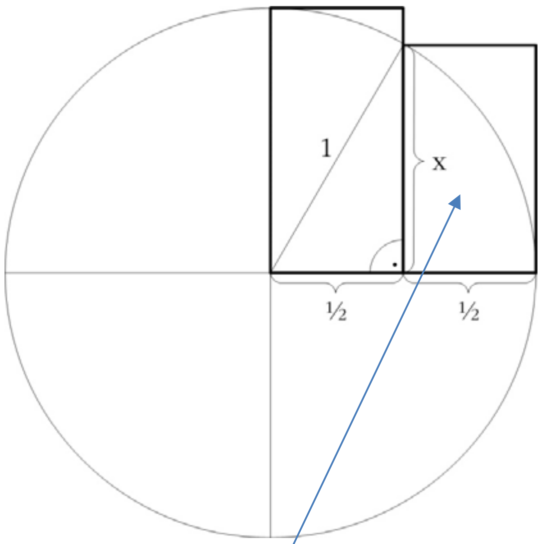
Eine weitere Näherung mit zwei Rechtecken.



Fläche linkes Rechteck $= 1 \cdot \frac{1}{2}$

Kreisfläche Rechtecknäherung (II)

Für das rechte Rechteck muss zunächst
dessen Höhe x berechnet werden.

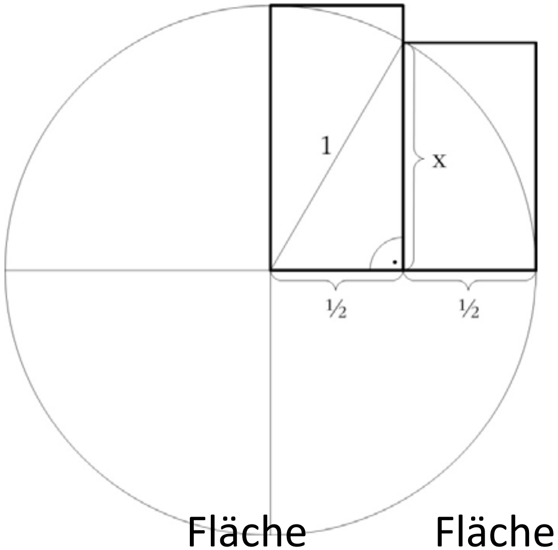


$$x^2 + \left(\frac{1}{2}\right)^2 = 1^2 \Leftrightarrow x^2 = \frac{3}{4}$$

$$\Rightarrow x = \sqrt{\frac{3}{4}}$$

Fläche rechtes Rechteck $= x \cdot \frac{1}{2} = \sqrt{\frac{3}{4}} \cdot \frac{1}{2}$

Kreisfläche Rechtecknäherung (III)



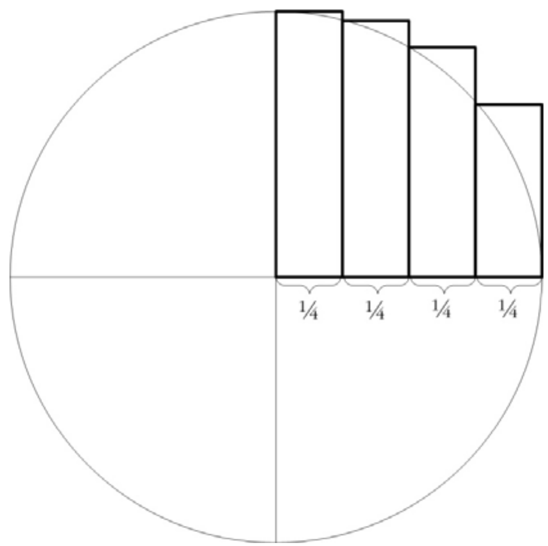
Fläche linkes Rechteck

Fläche rechtes Rechteck

$$\frac{\pi}{4} \approx \frac{1}{2} + x \cdot \frac{1}{2} = \frac{1}{2} + \sqrt{\frac{3}{4}} \cdot \frac{1}{2} \approx 0{,}93301$$

$$\pi \approx 3{,}73205$$

Aufgabe
π Rechtecknäherung



Hinweis für das Übungsblatt:
Berechnen Sie den Flächeninhalt zu obenstehender Zeichnung (auf Papier)!