

# Automatische Überprüfung der Anwendung mathematischer Näherungsverfahren durch Coderunner

18. Workshop Mathematik in ingenieurwissenschaftlichen Studiengängen

**Alexander Dominicus**

10.11.2023



# Table of Contents

Problembeschreibung

Coderunner: Erklärung und Beispiele

Coderunner: Benotung

# Studierende sollen den Umgang mit iterativen (mathematische) Verfahren erlernen

## Iterative Verfahren...

- näherungsweise Lösen einer Gleichung oder eines Gleichungssystems
- näherungsweise Darstellung von Zahlen
- numerische Lösen von (gewöhnlichen) Differentialgleichungen
- Approximation von Funktionen durch Polynome
- usw.

# Studierende sollen den Umgang mit iterativen (mathematische) Verfahren erlernen

## Iterative Verfahren...

- näherungsweise Lösen einer Gleichung oder eines Gleichungssystems
- näherungsweise Darstellung von Zahlen
- numerische Lösen von (gewöhnlichen) Differentialgleichungen
- Approximation von Funktionen durch Polynome
- usw.

## ...werden in verschiedenen Bereichen der Hochschullehre angewendet

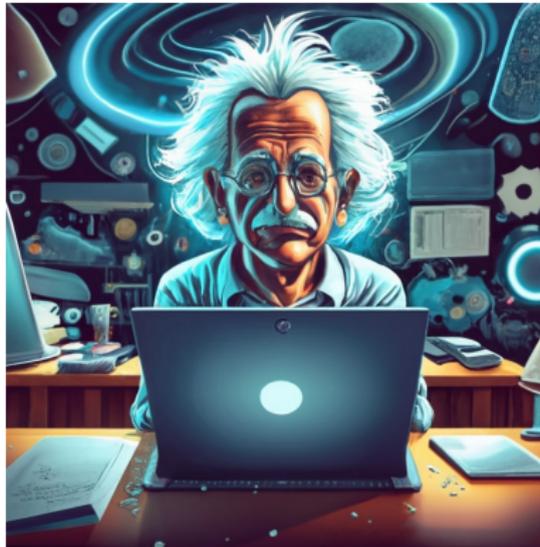
- Mathematik
- Ingenieurwissenschaften
- Informatik
- usw.

## Musteraufgabe

Erstellen Sie einen Python-Programmcode, der mit Hilfe des Heron/Newton-Verfahrens für beliebiges  $a \in \mathbb{N} > 0$  eine Approximation  $\text{my\_sqrt}(a)$  berechnet, sodass

$$|\text{my\_sqrt}(a) - \sqrt{a}| < 10^{-6}$$

gilt. Der Code sollte effizient sein d.h. so wenig Iterationen wie möglich verwenden.



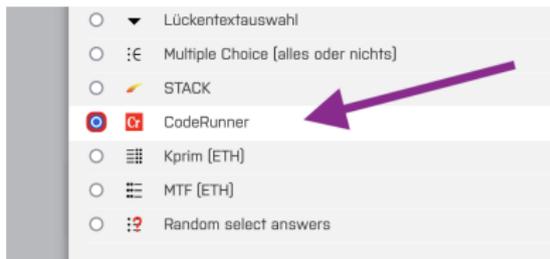
## verschiedene Problemlösungen

- **klassischer Ansatz:** Studierende erstellen Programmcode, Ausführung und Evaluation durch Dozierende/Mitarbeitende
  - ▷ zeitaufwändig
  - ▷ subjektiv in der Bewertung
- **automatisierter/modernen Ansatz:** Studierende erstellen Ihren Programmcode innerhalb eines Moodle Test und bekommen direkt eine objektive Bewertung Ihrer Leistung (inkl. Feedback)
- **Wie ist das möglich?**

# verschiedene Problemlösungen

- **klassischer Ansatz:** Studierende erstellen Programmcode, Ausführung und Evaluation durch Dozierende/Mitarbeitende
  - ▷ zeitaufwändig
  - ▷ subjektiv in der Bewertung
- **automatisierter/modernen Ansatz:** Studierende erstellen Ihren Programmcode innerhalb eines Moodle Test und bekommen direkt eine objektive Bewertung Ihrer Leistung (inkl. Feedback)
- **Wie ist das möglich?**

## CoderRunner-Plugin (Fragetyp):



# Table of Contents

Problembeschreibung

Coderunner: Erklärung und Beispiele

Coderunner: Benotung

# Was ist Coderunner?

- **Moodle Fragetyp**
- **Studierende erstellen Programmcode**
  - ▷ Code wird auf Fehler ausgeführt und auf Fehler geprüft (Jobe-Server)
  - ▷ Ausgabe des Codes wird automatisch bewertet
- **Unterschiedliche Programmiersprachen unterstützt (Python3, C, C++, Java, PHP, JavaScript (NodeJS), SQL, Octave/Matlab)**



# Beispiel I: Octave

Schreiben Sie eine Octave-Funktion `quadriere(x)`, welche die Quadratzahl von  $x \in \mathbb{R}$  berechnet.

Zum Beispiel:

Test	Resultat
<code>disp(quadriere(7))</code>	49
<code>disp(quadriere(-11))</code>	121

Antwort: [Abzugssystem: 33.3, 66.7, ... %]

```

1 function sq=quadriere(x)
2     sq = x * x;
3 end
    
```

Prüfen

	Test	Erwartet	Erhalten	
✓	<code>disp(quadriere(7))</code>	49	49	✓
✓	<code>disp(quadriere(-11))</code>	121	121	✓
✓	<code>disp(quadriere(0))</code>	0	0	✓
✓	<code>disp(quadriere(-101))</code>	10201	10201	✓
✓	<code>disp(quadriere(20))</code>	400	400	✓

Alle Tests bestanden! ✓

Abbildung: Beispiel: Octave-Programmcode

# Beispiel II: Python

Schreiben Sie ein Python-Programm `print_sqrs_1_to_n(n)`, welches tabellarisch die Quadrate einer Zahl  $n \in \mathbb{N}$  auflistet.

Zum Beispiel:

Test	Resultat
<code>print_sqrs_1_to_n(5)</code>	<pre>1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25</pre>

Antwort: [Abzugssystem: 10, 20, ... %]

```

1 def print_sqrs_1_to_n(n):
2     for i in range(1,n):
3         quad=i*i
4         print(str(i)+" * "+str(i)+" = "+str(i*i))
5     print(str(n)+" * "+str(n)+" = "+str(n*n))
    
```

Prüfen

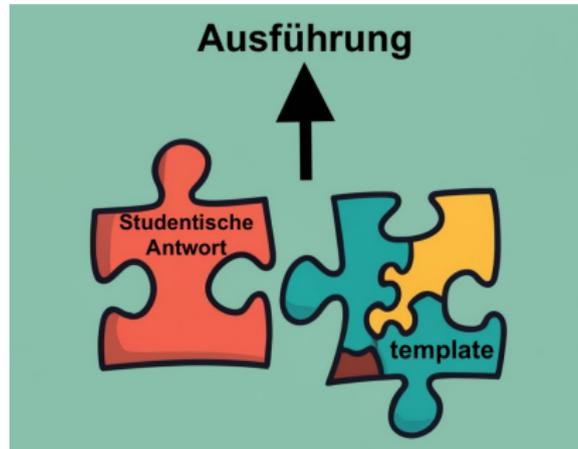
Test	Erwartet	Erhalten	
✓ <code>print_sqrs_1_to_n(2)</code>	<pre>1 * 1 = 1 2 * 2 = 4</pre>	<pre>1 * 1 = 1 2 * 2 = 4</pre>	✓
✓ <code>print_sqrs_1_to_n(5)</code>	<pre>1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25</pre>	<pre>1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25</pre>	✓
✓ <code>print_sqrs_1_to_n(11)</code>	<pre>1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25 6 * 6 = 36 7 * 7 = 49 8 * 8 = 64 9 * 9 = 81 10 * 10 = 100 11 * 11 = 121</pre>	<pre>1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25 6 * 6 = 36 7 * 7 = 49 8 * 8 = 64 9 * 9 = 81 10 * 10 = 100 11 * 11 = 121</pre>	✓
✓ <code>print_sqrs_1_to_n(0)</code>	<pre>0 * 0 = 0</pre>	<pre>0 * 0 = 0</pre>	✓

Alle Tests bestanden! ✓

Abbildung: Beispiel: Python-Programmcode

## Codeausführung und Benotung: *Exakte Übereinstimmung*

1. Der studentische Programmcode wird in einen *template* eingebettet und damit ausführbar.
2. Der erweiterte Programmcode wird ausgeführt und bewertet:
  - ▷ Grundlage der Bewertung bei den gegebenen Beispielen ist die **exakte Übereinstimmung** der Studentischen Antwort mit der hinterlegten Antwort
  - ▷ Nur bei exakter Übereinstimmung wird die Frage als richtig gewertet, sonst als falsch



# Bewertung: *Exakte Übereinstimmung*

Test	Erwartet	Erhalten	
✓ print_sqrs_1_to_n(2)	1 * 1 = 1 2 * 2 = 4	1 * 1 = 1 2 * 2 = 4	✓
✓ print_sqrs_1_to_n(5)	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	✓
✓ print_sqrs_1_to_n(11)	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25 6 * 6 = 36 7 * 7 = 49 8 * 8 = 64 9 * 9 = 81 10 * 10 = 100 11 * 11 = 121	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25 6 * 6 = 36 7 * 7 = 49 8 * 8 = 64 9 * 9 = 81 10 * 10 = 100 11 * 11 = 121	✓
✓ print_sqrs_1_to_n(0)	0 * 0 = 0	0 * 0 = 0	✓

Alle Tests bestanden! ✓

Test	Erwartet	Erhalten	
✗ print_sqrs_1_to_n(2)	1 * 1 = 1 2 * 2 = 4	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	✗
✓ print_sqrs_1_to_n(5)	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	✓
✗ print_sqrs_1_to_n(11)	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25 6 * 6 = 36 7 * 7 = 49 8 * 8 = 64 9 * 9 = 81 10 * 10 = 100 11 * 11 = 121	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	✗
✗ print_sqrs_1_to_n(0)	0 * 0 = 0	1 * 1 = 1 2 * 2 = 4 3 * 3 = 9 4 * 4 = 16 5 * 5 = 25	✗

Ihr Code muss alle Tests bestehen, um eine Bewertung zu erhalten. Versuchen Sie es noch einmal.  
Unterschiede anzeigen

Abbildung: Vergleich: korrekt und falsch gewertete Antworten

## Zurück zur Musteraufgabe...

### Musteraufgabe

Erstellen Sie einen Python-Programmcode, der mit Hilfe des Heron/Newton-Verfahrens für beliebiges  $a \in \mathbb{N} > 0$  eine Approximation `my_sqrt(a)` berechnet, sodass

$$|\text{my\_sqrt}(a) - \sqrt{a}| < 10^{-6}$$

gilt. Der Code sollte effizient sein d.h. so wenig Iterationen wie möglich verwenden.

**Musteraufgabe ↔ exakte Übereinstimmung ⚡**

## Zurück zur Musteraufgabe...

### Musteraufgabe

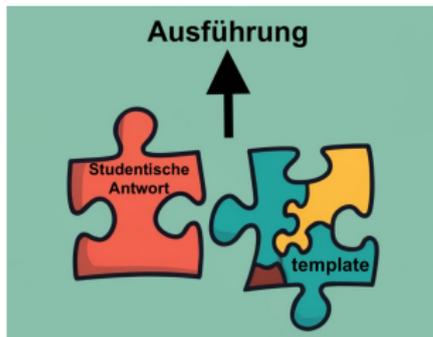
Erstellen Sie einen Python-Programmcode, der mit Hilfe des Heron/Newton-Verfahrens für beliebiges  $a \in \mathbb{N} > 0$  eine Approximation  $\text{my\_sqrt}(a)$  berechnet, sodass

$$|\text{my\_sqrt}(a) - \sqrt{a}| < 10^{-6}$$

gilt. Der Code sollte effizient sein d.h. so wenig Iterationen wie möglich verwenden.

**Musteraufgabe** ↔ **exakte Übereinstimmung** ⚡

**Lösung:**



## Zurück zur Musteraufgabe...

### Musteraufgabe

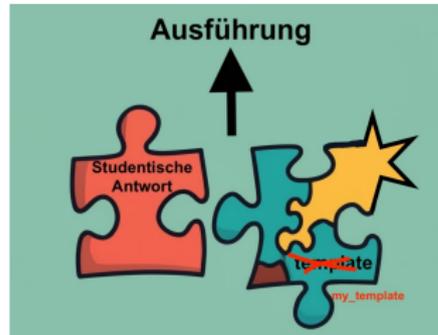
Erstellen Sie einen Python-Programmcode, der mit Hilfe des Heron/Newton-Verfahrens für beliebiges  $a \in \mathbb{N} > 0$  eine Approximation  $\text{my\_sqrt}(a)$  berechnet, sodass

$$|\text{my\_sqrt}(a) - \sqrt{a}| < 10^{-6}$$

gilt. Der Code sollte effizient sein d.h. so wenig Iterationen wie möglich verwenden.

**Musteraufgabe** ↔ **exakte Übereinstimmung** ⚡

**Lösung:**



# Table of Contents

Problembeschreibung

Coderunner: Erklärung und Beispiele

Coderunner: Benotung

# Anpassung der Benotung

- Coderunner bietet die Möglichkeit das template anzupassen in den der studentische Programmcode eingebettet wird
  - ▷ **Bewertung/Benotung** anpassen
  - ▷ spezifisches **Feedback**
  - ▷ Antworten werten, die keine exakte Übereinstimmung benötigen
- **Musteraufgabe kann in Moodle abgeprüft werden!**

## CodeRunner-Fragetyp

Fragetyp

Anpassung  Anpassen  Vorlagen-Debugging

Antwortfeld  Hilfe für Anpassung

Prüfen-Buttons  Vorabprüfung   Prüfen-Button

Stop Button  Nie verfügbar

Feedback

Bewertung  Alles-oder-nichts-Bewertung  Abzugssystem:

Anpassung

Vorlage

```
1 {{ STUDENT_ANSWER }}
2 ____student_answer____ = ""({{ STUDENT_ANSWER | e("py") }})""
3
4 SEPARATOR = "#####"
5
6
7 {% for TEST in TESTCASES %}
8 {{ TEST | include: 2 }}
9 {% if not Loop.Last %}
10 print(SEPARATOR)
11 {% endif %}
12 {% endifor %}
```

Vorlagensteuerung  Ist ein Kombinator  Mehrere allow erlauben  
TestTransier (logged) | f4ab0179430f10f0-0-Peiris

Benotung  Genaue Übereinstimmung

Abbildung: Bewertung Anpassen bei Coderunner Fragetyp



# Download der Beispiele

Alle hier gezeigten Beispiele finden Sie zum Download auf der Homepage des DigiTeach-Instituts

<https://www.hochschule-bochum.de/digiteach/materialien-downloads/>

## Download von Materialien

### Vorträge & Veröffentlichungen

 Foliien zum ChatGPT-Workshop (L^3/W^3) am 8.3.2023	PDF   1 MB
 Foliien zum ChatGPT-Vortrag im Rahmen des HDW-Mentor:innen-Treffens	PDF   722 KB
 Poster zum Workshoptag "Innovative digitale Werkzeuge in Studium und Lehre"	PDF   551 KB
 Template_Aufbau_einer_digitalen_Lerneinheit__ELE_final.pdf	PDF   337 KB
 Was_ist_E-Learning_und_wann_ist_es_erfolgreich_final.pdf	PDF   229 KB

### Lehr und Lernmaterialien

 CCD-Icons	ZIP   23 MB
 Trailer Kurzfilm-Serie - "Die Unternehmensbrater" (Edutainment made @BO)	YOUTUBE   11 B
 CodeRunner Beispiele	ZIP   9 KB



Ende: Zeit für Diskussionen!

**Vielen Dank für Ihre Aufmerksamkeit!**  
Gibt es Fragen?



**Email:** [alexander.dominicus@hs-bochum.de](mailto:alexander.dominicus@hs-bochum.de)