



# Informatik I

## Übungsstunde 11

Herbst 2020

# Revision Aufgaben

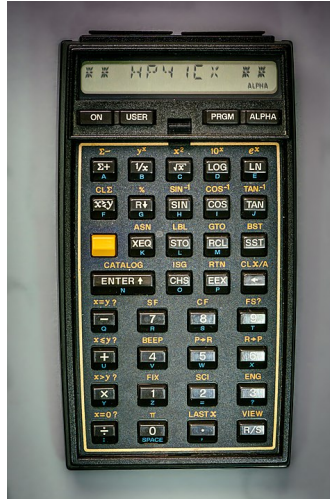
- Polymorphic Animals
- Polymorphism and Overrides
- Mr Brush
- Comparing Rational Numbers

# Beispiel

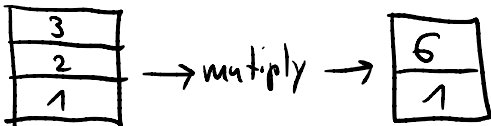
Taschenrechner mit umgekehrter polnischer Notation.

Auf English: *Reverse Polish Notation (RPN)*.  
Ein Taschenrechner mit einem Stack.

# RPN-Rechner



# RPN-Rechner

Unterstützte Operationen: 

- **push(value)** Pushe einen Wert auf den Stack
- **pop()** Poppe einen Wert vom Stack und zeige den Wert an
- **add(), subtract(), multiply(), divide()** Grundlegende arithmetische binäre Operationen: Poppe die Werte  $v_2$  und  $v_1$  vom Stack und pushe  $v_2$  **op**  $v_1$

# Stack

# RPN-Rechner

$$(4 \cdot 2) - (5 + 1) / 2 = 5$$

2
4

# RPN-Rechner

$$(4 \cdot 2) - (5 + 1) / 2 = 5$$

multiply()

2
4

2
4



# RPN-Rechner

$$(4 \cdot 2) - (5 + 1) / 2 = 5$$

multiply()

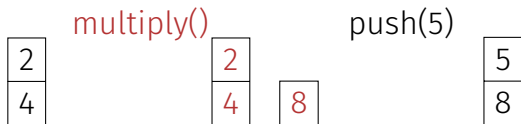
2
4

2
4

8
---

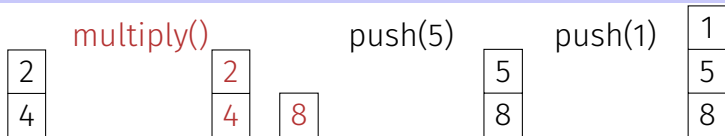
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1) / 2 = 5$$



# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



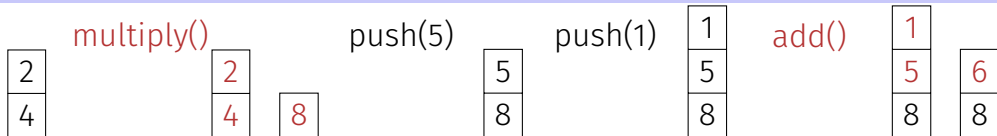
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



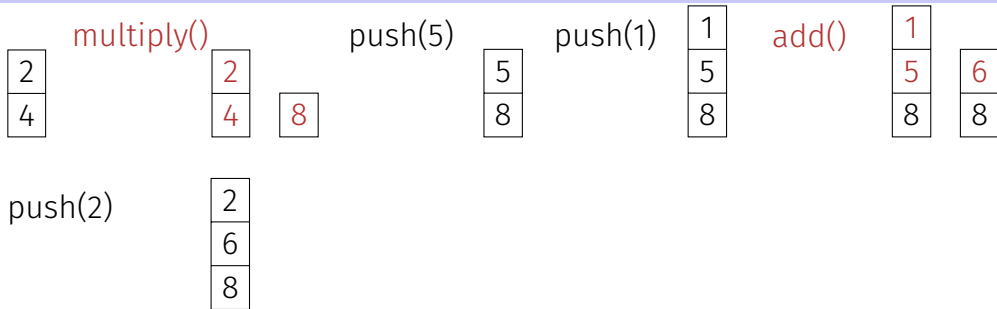
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



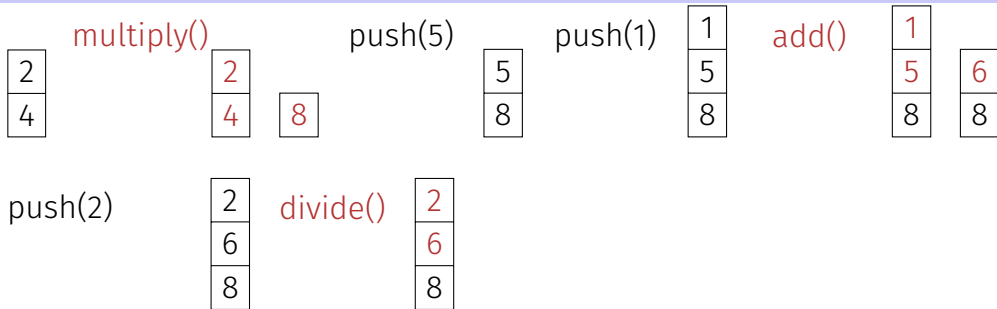
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



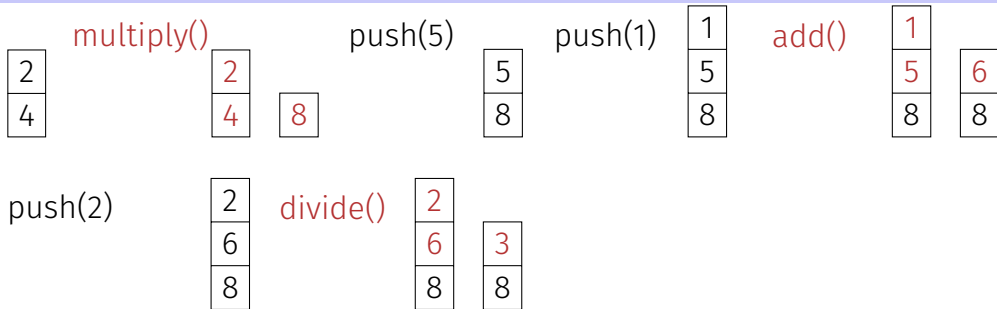
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



# RPN-Rechner

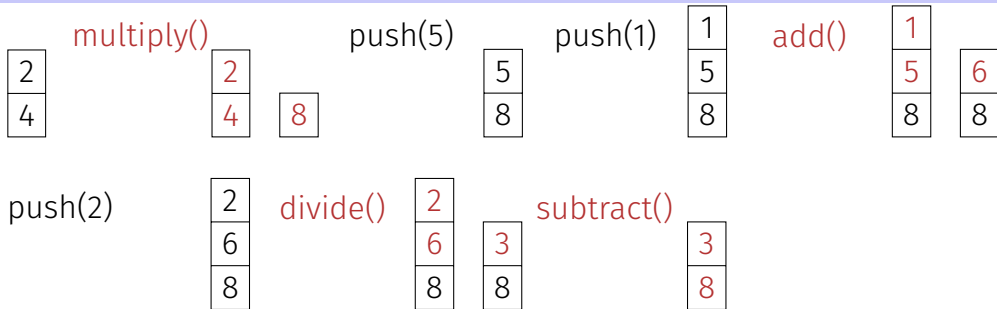
$$(4 \cdot 2) - (5 + 1)/2 = 5$$





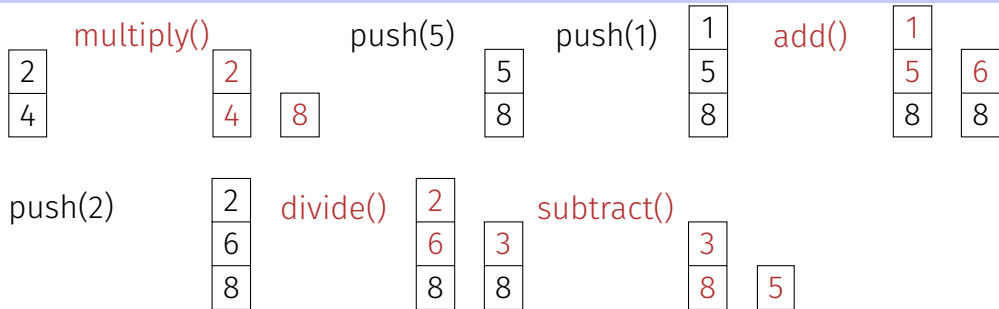
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



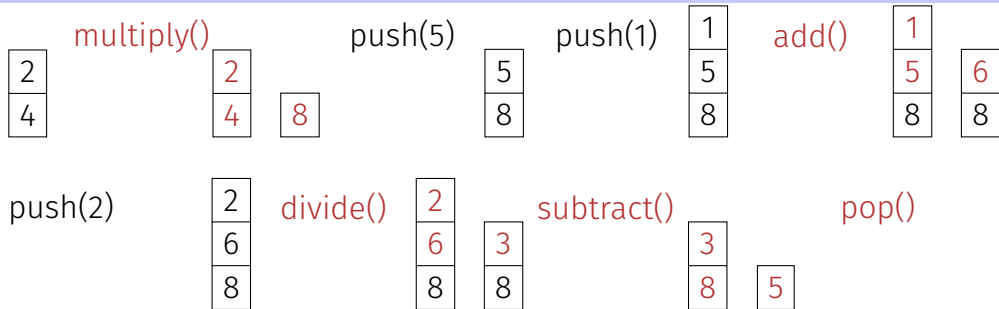
# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



# RPN-Rechner

$$(4 \cdot 2) - (5 + 1)/2 = 5$$



# RPN-Rechner

$$(4 \cdot 2) - (5 + 1) / 2 = 5$$

4 2 \* 5 1 + 2 / - =

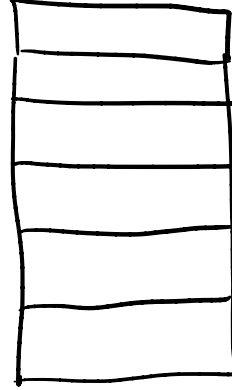
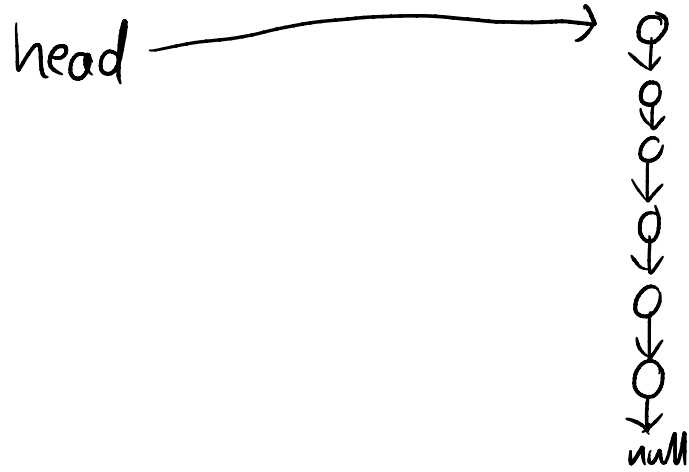
Commands:

- value  $\rightarrow$  push(value)
- =  $\rightarrow$  pop()
- +, -, \*, /  $\rightarrow$  add(), subtract(), multiply(), divide()

# RPN-Rechner

Implementiere die Stack-Datenstruktur für den Rechner.

Stack.



# Vorschau Aufgaben

Über **dynamische Datenstrukturen**.

- Umgekehrte Ausgabe
- Invarianten der Datenstruktur Queue
- Implementation einer Warteschlange (Queue)
- Wörterbuch

# Invarianten

"Eine Invariante ist eine Aussage, die über die Ausführung bestimmter Programmbeefehle Garantien gibt".

```
x > 0  


{code}

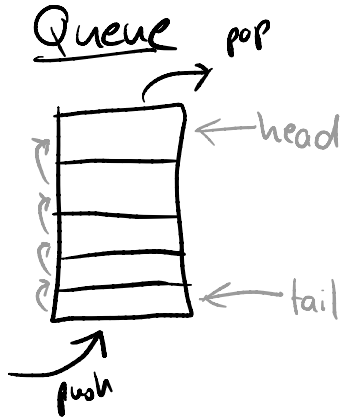
  
x > 0  
  
for (...) {  
    i > 0  


{code}

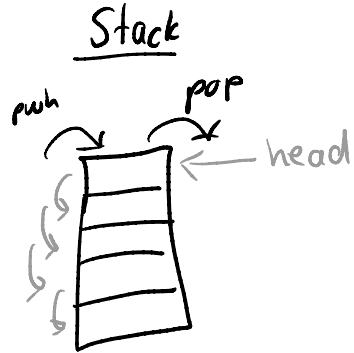
  
}
```



# Queue (Warteschlange)



vs.



# Bonusaufgaben

- Caesar Verschlüsselung
- Automatic Differentiation