

# LU10.L03 - Einleser erweitern

## Aufgaben

### Teilaufgabe 1

Nehmen Sie die Github-Classroom Aufgabe an und clonen Sie das Repository in ihre Entwicklungsumgebung.

Studieren Sie die zwei Python Files `input_reader.py` und `main.py`.

- Sehen Sie im File `main.py` an welcher stelle die Grenzwerte mitgegeben werden? Vielleicht erinnern Sie sich an die [LU09](#) wo wir die benannten Argumente kennengelernt haben.
- Sehen Sie im File `input_reader.py` das wir die Paramter für die Funktion noch nicht mit `lower_bound` und `upper_bound` ergänzt haben? In der LU09 haben wir auch kennengelernt wie sie Default-Werte für Parameter definieren. Dieses wissen werden wir hier jetzt benötigen.

### Teilauftrag 2

None

```
def read_float(text_to_display, lower_bound =
None, upper_bound = None):
...

def read_int(text_to_display, lower_bound =
None, upper_bound = None):
...
```

**math.inf**

```
def read_float(text_to_display, lower_bound =
float(-inf), upper_bound = float(inf)):
...

def read_int(text_to_display, lower_bound =
int(-inf), upper_bound = int(inf)):
...
```

## Teilauftrag 3

Überlegen Sie sich, wie Sie die Funktion `read_float(text_to_display, lower_bound = XX, upper_bound = XX)`: anpassen müssen, damit die Überprüfung auf die Grenzwerte funktioniert.

**None**

```
def read_float(text_to_display, lower_bound =
None, upper_bound = None):
    """
    Read a float from the user within bounds
    """
    while True:
        try:
            num =
float(input(text_to_display))
        except ValueError:
            print("Please, enter a real
number!")
            continue
        else:
            if lower_bound is not None and
num < lower_bound:
                print("Please, enter a number
greater than or equal to", lower_bound)
                continue
            if upper_bound is not None and
num > upper_bound:
                print("Please, enter a number
less than or equal to", upper_bound)
                continue
            return num

def read_int(text_to_display, lower_bound =
None, upper_bound = None):
    """
    Read an int from the user within bounds
    """
    while True:
        try:
            num = int(input(text_to_display))
        except ValueError:
            print("Please, enter a whole
number!")
            continue
        else:
            if lower_bound is not None and
```

```
num < lower_bound:
    print("Please, enter a number
greater than or equal to", lower_bound)
    continue
    if upper_bound is not None and
num > upper_bound:
    print("Please, enter a number
less than or equal to", upper_bound)
    continue
    return num
```

### math.inf

```
from math import inf

def read_float(text_to_display, lower_bound =
float(-inf), upper_bound = float(inf)):
    """
    Read a float from the user within bounds
    """
    while True:
        try:
            num =
float(input(text_to_display))
        except ValueError:
            print("Please, enter a real
number!")
            continue
        else:
            if num < lower_bound:
                print("Please, enter a number
greater than or equal to", lower_bound)
                continue
            if num > upper_bound:
                print("Please, enter a number
less than or equal to", upper_bound)
                continue
            return num

def read_int(text_to_display, lower_bound =
float(-inf), upper_bound = float(inf)):
    """
    Read an int from the user within bounds
    """
    while True:
        try:
            num = int(input(text_to_display))
        except ValueError:
            print("Please, enter a whole
number!")
```

```
        continue
    else:
        if num < lower_bound:
            print("Please, enter a number
greater than or equal to", lower_bound)
            continue
        if num > upper_bound:
            print("Please, enter a number
less than or equal to", upper_bound)
            continue
    return num
```

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

<https://wiki.bzz.ch/de/modul/m319/learningunits/lu10/loesungen/einlesererweitert>

Last update: **2025/06/23 07:45**

