Herzlich Willkommen bei Inf-Einf-B Woche 6.

Heute sehen wir uns **Python** an. Dies wird in unserem Kurs eine von zwei Vorlesungen sein, in der wir Python an sich behandeln.

Es gibt Übungen und Shorts zu weiterführenden Konstrukten von Python.

Nächste Vorlesungen:
22.12.25 zu Python (Fortsetzung)

12.01.26 zu Web: HTML/CSS/JavaScript (Client)

19.01.26 zu Web: Python am Server (Flask)

Langsam auch relevant:

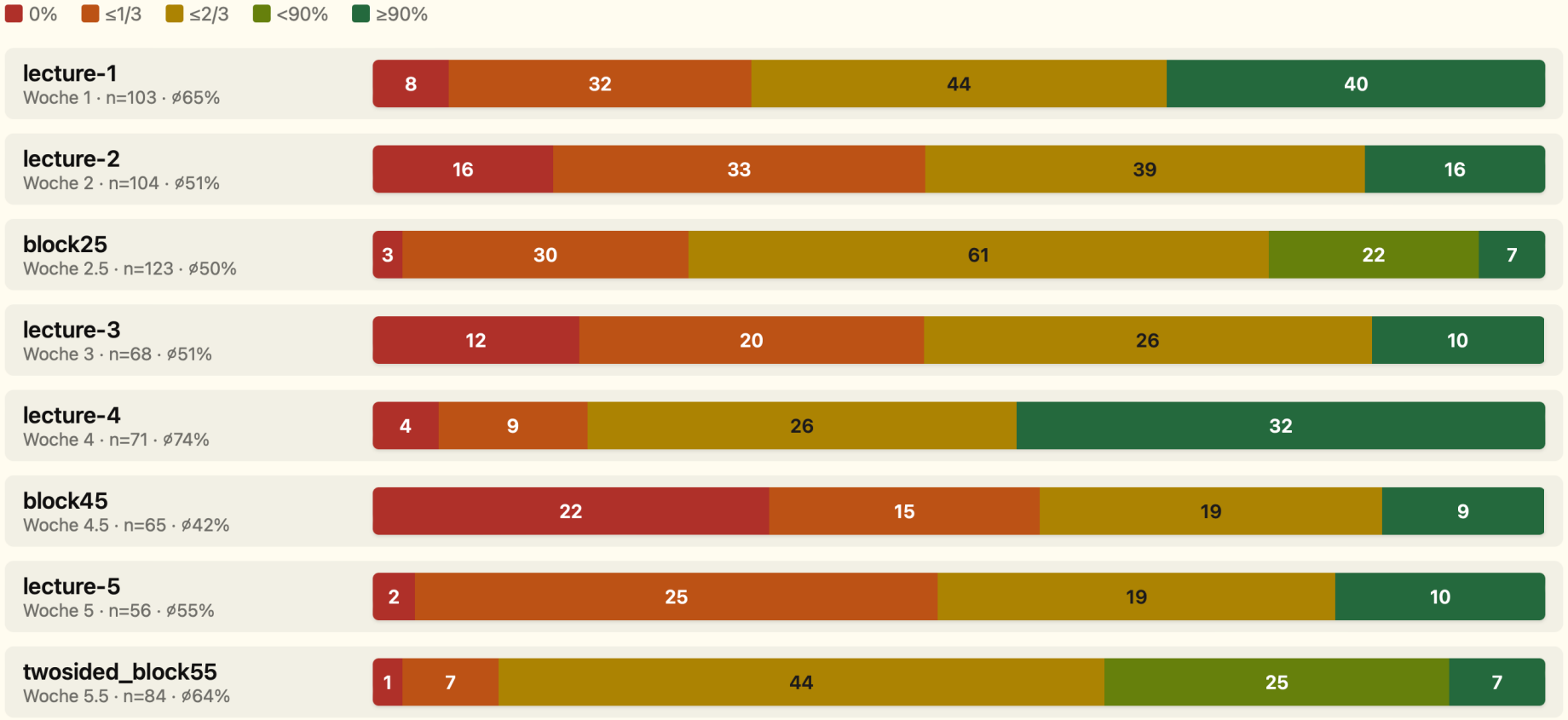
Ideen für Abschlussprojekt erarbeiten und gerne mit mir oder Tutoren besprechen.

Hackathon
am 30.1.26 nachmittags/nachts

Vorstellung auf Messe
am 2.2.26.

Dies ist Inf-Einf-B.

Kurs-Statistiken



PROBLEM 1

Aufgabe: Füllen Sie die zwei Lücken aus, sodass das Programm eine Textdatei zeilenweise einliest und ausgibt

```
#include <stdio.h>

int main(void)
{
    FILE *f = fopen("namen.txt", "r");
    if (f == NULL) return 1;

    char zeile[100];
    while (___(1)___ != NULL)
        printf("Gelesen: %s", ___(2)___);

    fclose(f);
}
```

Optionen für (1):
fgets(zeile, 100, f), fgets(f, 100, zeile), fgets(f, zeile, 100), fgetc(f)

Optionen für (2): f, zeile, &zeile, *zeile

PROBLEM 1

```c
#include <stdio.h>

int main(void)
{
    FILE *f = fopen("namen.txt", "r");
    if (f == NULL) return 1;

    char zeile[100];
    while(fgets(zeile, 100, f) != NULL)
        printf("Gelesen: %s", zeile);

    fclose(f);
}
```

Aufgabe: Füllen Sie die zwei Lücken aus, sodass das Programm eine Textdatei zeilenweise einliest und ausgibt

Optionen für (1):
fgets(zeile, 100, f), fgets(f, 100, zeile), fgets(f, zeile, 100), fgetc(f)

Optionen für (2): f, zeile, &zeile, *zeile

PROBLEM 2

```
typedef struct {
    char name[20];
    int score;
} player;

player alice = {"Alice", 100};
player *ptr = &alice;
```

Welche Zugriffe sind korrekt? (✓ oder –)

```
printf("%s", alice.name);      // ?
printf("%d", alice->score);    // ?
printf("%s", ptr.name);        // ?
printf("%d", ptr->score);      // ?
printf("%d", (*ptr).score);    // ?
```

PROBLEM 2

```
typedef struct {
    char name[20];
    int score;
} player;

player alice = {"Alice", 100};
player *ptr = &alice;
```

Welche Zugriffe sind korrekt? (✓ oder –)

```
printf("%s", alice.name);      // ✓
printf("%d", alice->score);    // –
printf("%s", ptr.name);        // –
printf("%d", ptr->score);      // ✓
printf("%d", (*ptr).score);    // ✓
```


when clicked
say
hello, world

```c
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
}
```

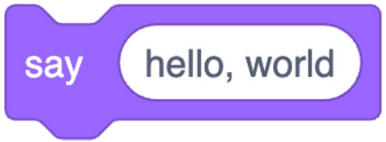
```
print("hello, world")
```

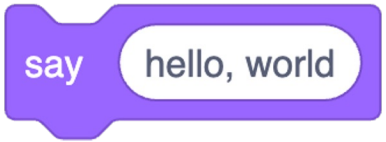
```
make hello

./hello
```

```
clang -o hello hello.c -lcs50

./hello
```

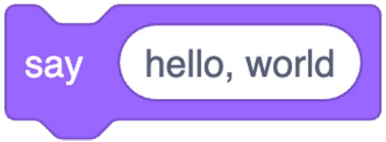
```
python hello.py
```

Funktionen


say
hello, world




```
printf("hello, world\n");
```



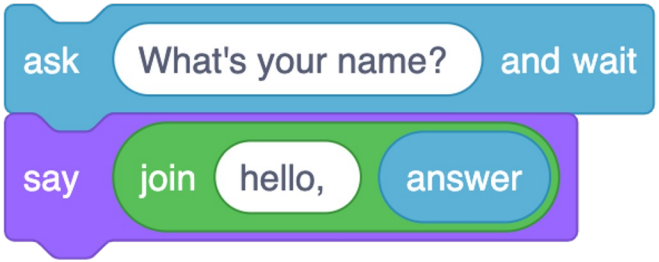

```
print("hello, world")
```

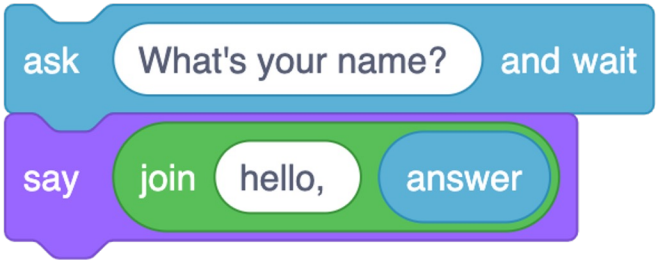
Bibliotheken

```
#include <cs50.h>
```

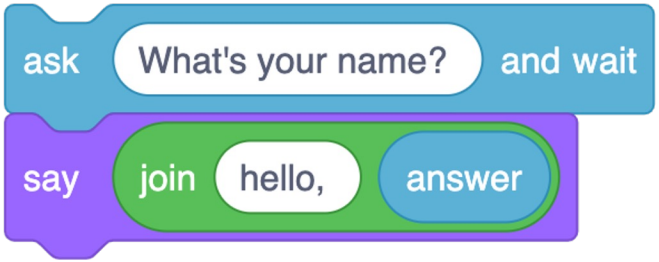
```
import cs50
```

```
from cs50 import get_string
```

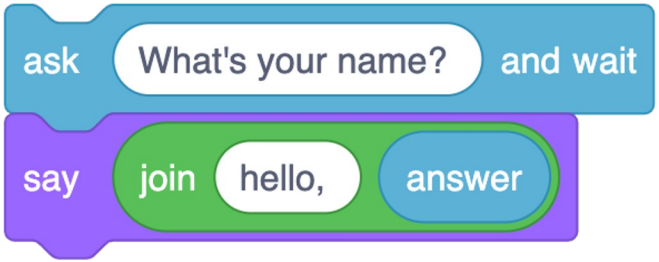

ask
What's your name?
and wait
say
join
hello,
answer




```
string answer = get_string("What's your name? ");
printf("hello, %s\n", answer);
```

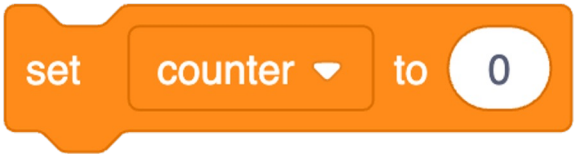


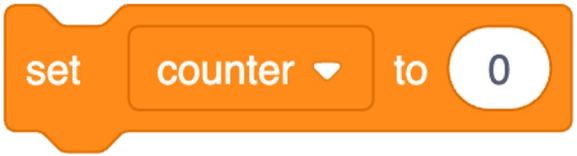

```
answer = get_string("What's your name? ")
print(f"hello, {answer}")
```



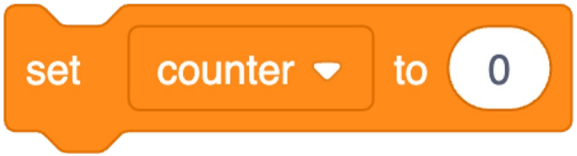

```
answer = input("What's your name? ")
print(f"hello, {answer}")
```

Variablen

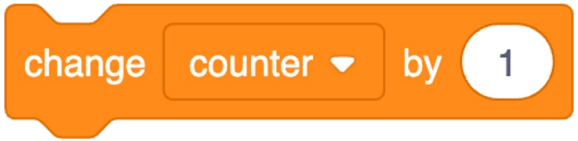

set
counter
to
0

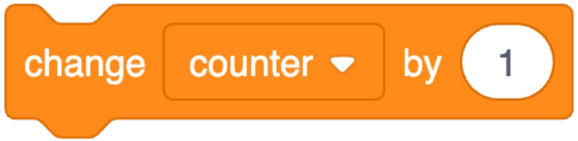



```
int counter = 0;
```

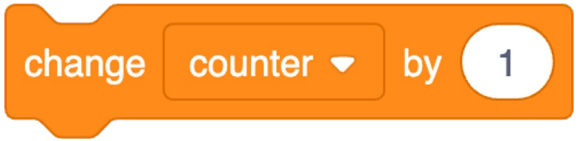



```
counter = 0
```

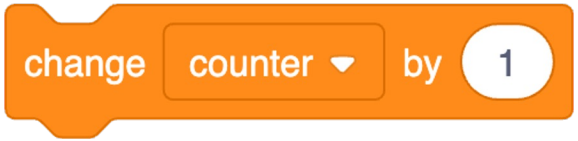

change
counter
by
1




```
counter = counter + 1;
```




```
counter = counter + 1
```




```
counter += 1
```

Datentypen

```
bool

char

double

float

int

long

string

...
```

```
bool



float
int


str
...
```

range

list

tuple

dict

set

...

```
get_char

get_double

get_float

get_int

get_long

get_string

...
```

```
get_float

get_int


get_string
```

```
from cs50 import get_float
from cs50 import get_int
from cs50 import get_string
```

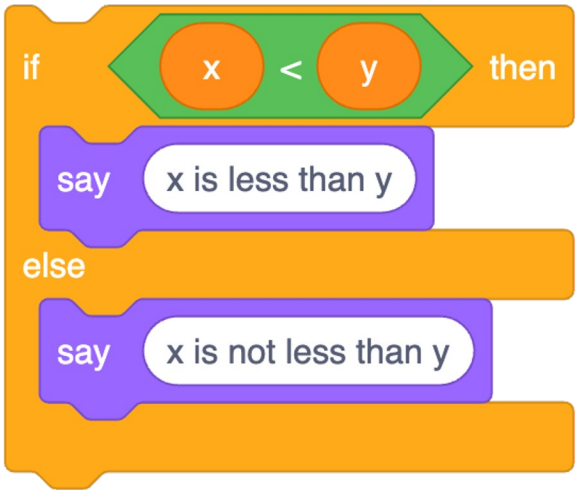
```
from cs50 import get_float, get_int, get_string
```

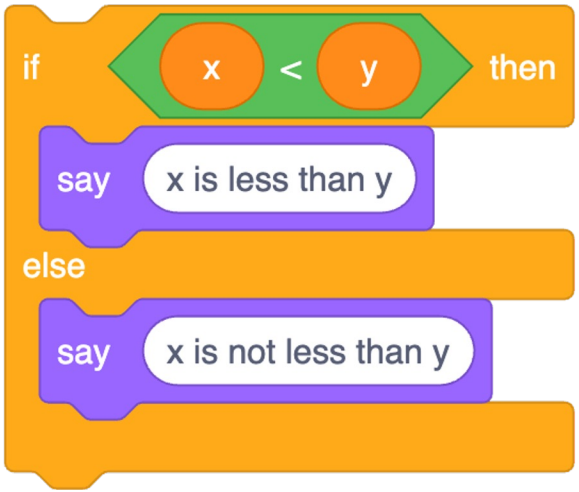
Bedingte Anweisungen


if
x
<
y
then
say
x is less than y

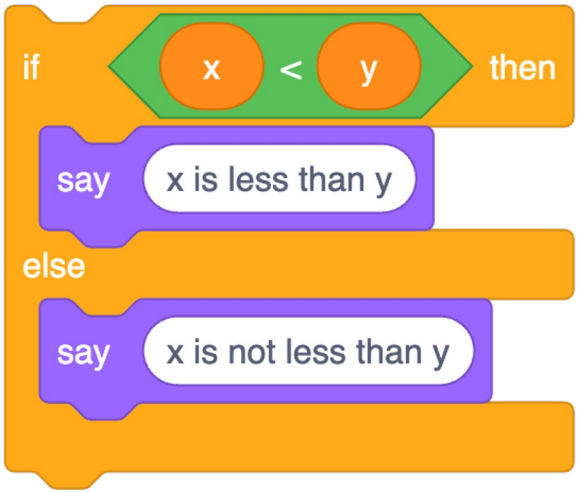

```
if (x < y)
{
    printf("x is less than y\n");
}
```


```
if x < y:
    print("x is less than y")
```

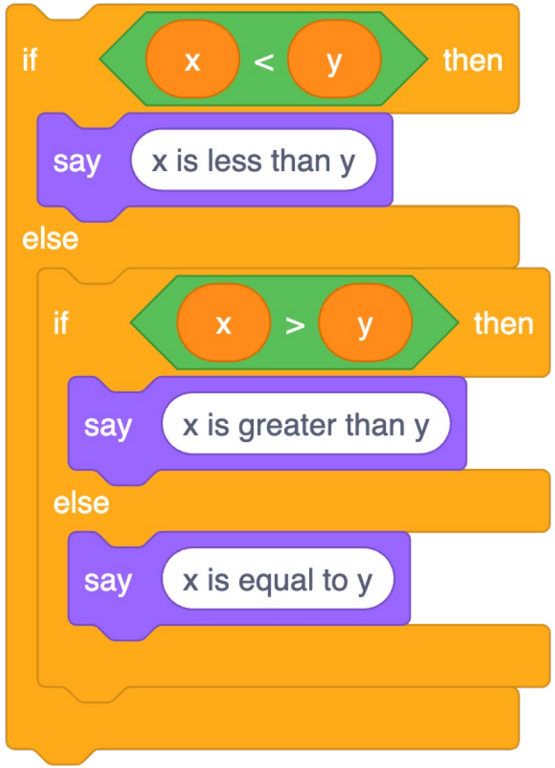

if
x
<
y
then
say
x is less than y
else
say
x is not less than y

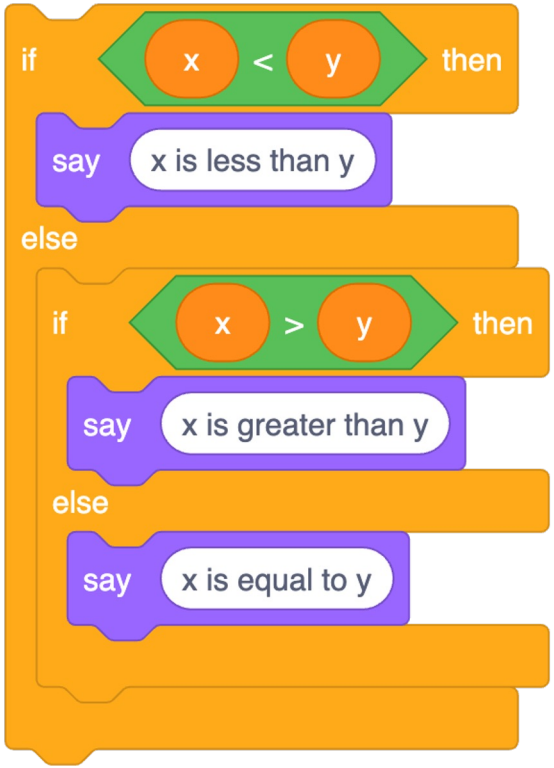



```
if (x < y)
{
    printf("x is less than y\n");
}
else
{
    printf("x is not less than y\n");
}
```

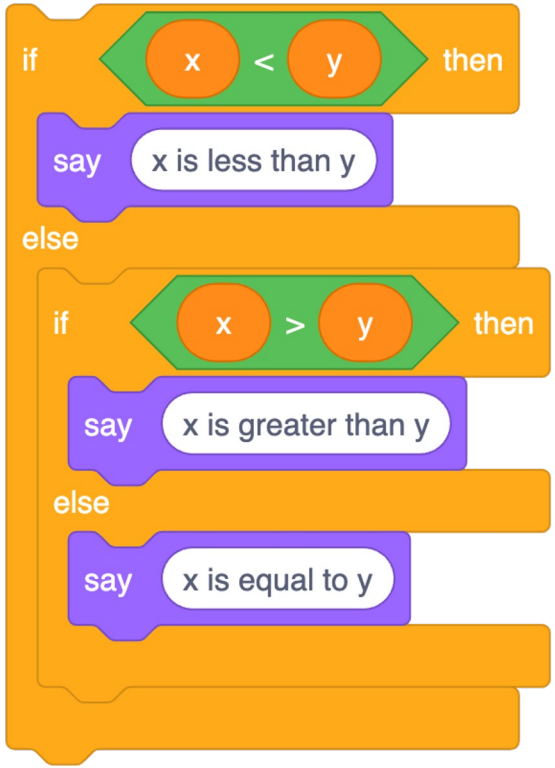



```
if x < y:
    print("x is less than y")
else:
    print("x is not less than y")
```


if
x
<
y
then
say
x is less than y
else
if
x
>
y
then
say
x is greater than y
else
say
x is equal to y




```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
    printf("x is equal to y\n");
}
```




```
if x < y:
    print("x is less than y")
elif x > y:
    print("x is greater than y")
else:
    print("x is equal to y")
```

str

PROBLEM 3

Aufgabe: Bringe diese Zeilen in
die richtige Reihenfolge, um einen
Knoten am Anfang einer Liste einzufügen:

```
// Gegeben: node *list zeigt auf bestehende Liste
// Gegeben: int value = 42

A: list = n;
B: n->next = list;
C: node *n = malloc(sizeof(node));
D: n->number = value;
```

Frage: Welche Reihenfolge ist korrekt?

PROBLEM 3

Aufgabe: Bringe diese Zeilen in
die richtige Reihenfolge, um einen
Knoten am Anfang einer Liste einzufügen:

```
// Gegeben: node *list zeigt auf bestehende Liste
// Gegeben: int value = 42

A: list = n;
B: n->next = list;
C: node *n = malloc(sizeof(node));
D: n->number = value;
```

Frage: Welche Reihenfolge ist korrekt?

C – D – B – A

PAUSE
20 min

Objekt-Orientiertes Programmieren

OOP

docs.python.org/3/library/stdtypes.html#string-methods

docs.python.org/3/library/functions.html

docs.python.org

Schleifen


repeat
3
say
meow


```
int i = 0;
while (i < 3)
{
    printf("meow\n");
    i++;
}
```


```
i = 0
while i < 3:
    print("meow")
    i += 1
```

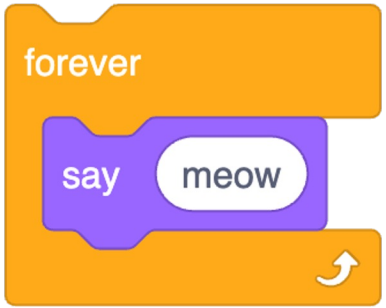

repeat
3
say
meow

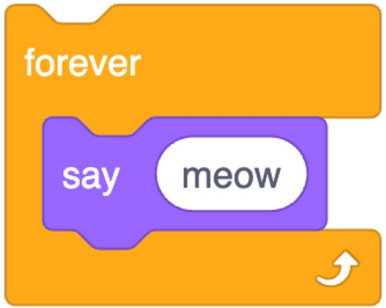

```
for (int i = 0; i < 3; i++)
{
    printf("meow\n");
}
```


```
for i in [0, 1, 2]:
    print("hello, world")
```

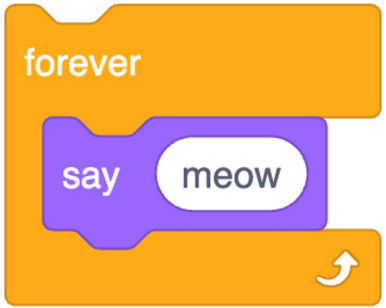

```
for i in range(3):
    print("hello, world")
```


```
for _ in range(3):
    print("hello, world")
```


forever
say
meow



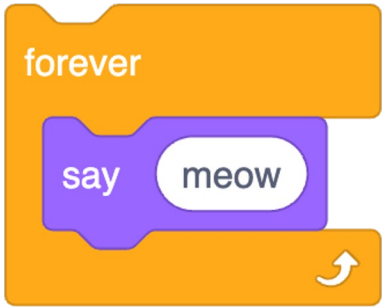

```
while (true)
{
    printf("meow\n");
}
```




```
while True:
    print("meow")
```

Benannte Parameter

Bisher: Positions-basierte Parameter




```
while True:
    print("meow")
```

Truncation

Ungenauigkeit von Fließkommazahlen

Integer Overflow

~~Integer Overflow~~

Exceptions

list

docs.python.org/3/library/stdtypes.html#sequence-types-list-tuple-range

len

docs.python.org/3/library/functions.html#len

dict

Schlüssel	Wert

docs.python.org/3/library/stdtypes.html#mapping-types-dict

Dies war Inf-Einf-B.