# Funktionen ohne Parameter

Eine Funktion ist ein ausgelagertes Unterprogramm, das beliebig oft im Hauptprogramm aufgerufen werden kann. Dadurch wird das ganze Programm übersichtlicher und einfacher zu entwickeln. Bestimmte Vorgänge, die immer wieder benötigt werden, können so ausgelagert werden und von jeder beliebigen Programmstelle aus immer wieder aufgerufen werden, anstatt sie jedes Mal neu zu erstellen.

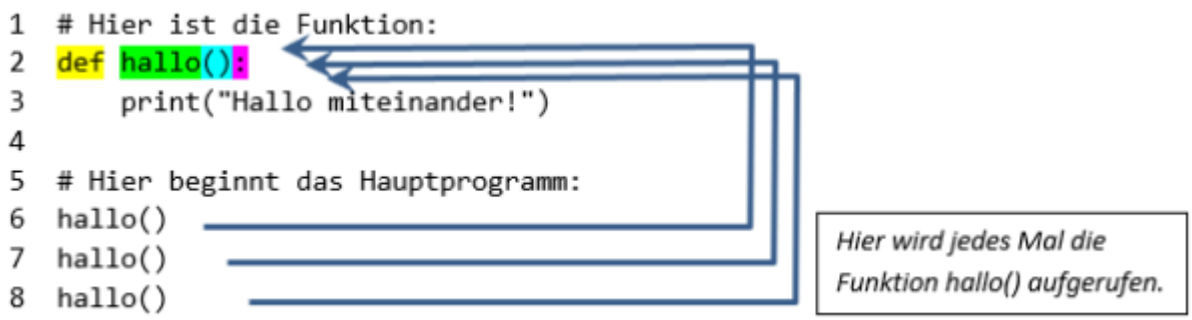
## Beispiel

<html> <iframe src=„https://trinket.io/embed/python/54028023f5“ width=„100%“ height=„300“ frameborder=„0“ marginwidth=„0“ marginheight=„0“ allowfullscreen></iframe> </html>

```
import turtle

# Das ist eine Funktion

def vor_und_zurueck():
  turtle.forward(100)
  turtle.right(180)
  turtle.forward(100)
  turtle.right(180)

# Hier beginnt das Hauptprogramm
vor_und_zurueck() #Aufruf der Funktion
vor_und_zurueck()
vor_und_zurueck()
```

- Die Definition der Funktion beginnt mit dem Schlüsselwort `def`. Danach folgt der `Name der Funktion`. (Für die Namensgebung gelten die gleichen Regeln wie für Variablen.)
- Anschließend folgen runde Klammern `()`, in denen sogenannte Funktionsparameter stehen können. Da hier im Beispiel keine Funktionsparameter übergeben werden, bleiben die Klammern leer.(In den folgenden Beispielen ändert sich das dann.)
- Hinter den Klammern folgt ein `Doppelpunkt`, um anzuzeigen, dass danach die Anweisungen folgen, mit denen die Funktion die Aufgaben durchführt, für die sie geschrieben wird. Im Beispiel hier sind das die Anweisungen:
  - turtle.forward(100)
  - turtle.right(180)
  - turtle.forward(100)
  - turtle.right(180)
- Wichtig: Die zur Funktion gehörigen Anweisungen müssen auf einer Ebene **eingerückt** sein.
- Im Hauptteil des Programms wird die Funktion vor_und_zurueck() dreimal aufgerufen. Der `Aufruf` besteht aus dem Namen der Funktion gefolgt von den runden Klammern. **Bei jedem Aufruf springt das Programm zur Funktion und führt sie aus**. **Nach der Ausführung der Funktion springt das Programm wieder zur Aufrufstelle zurück und führt die Anweisung aus, die auf den Funktionsaufruf folgt.** Auf diese Weise wird im vorliegenden Beispiel dreimal die Funktion vor_und_zurueck() aufgerufen.

# Veranschaulichendes Beispiel




# Allgemeine Syntax - Funktion ohne Parameter

Mit `def bezeichner():` definierst du einen neuen Befehl. Wähle einen Namen, der die Tätigkeit wiederspiegelt. Alle Anweisungen, die zum neuen Befehl gehören, müssen eingerückt sein.

```
def bezeichner():
    Anweisungen
```

Vergiss die **Klammern** und den **Doppelpunkt** nach dem Bezeichner nicht! In Python nennt man neue Befehle auch **Funktionen**. Wenn du die Funktion vor_und_zurueck() verwendest, sagt man auch, die Funktion werde **„aufgerufen“**.

Wir gewöhnen uns daran, die Funktionsdefinitionen im Programmkopf anzuordnen, da diese vor ihrem Aufruf definiert sein müssen.

```
####################
#Import von Modulen
#z.B.
import turtle


####################
#Funktionsblock
#Hier werden Funktionen definiert, z.B.
def vor_und_zurueck():
   turtle.forward(100)
   turtle.right(180)
   turtle.forward(100)
   turtle.right(180)


####################
#Hauptblock
#Hier folgt der Aufruf der Funktionen, z.B.
```

```
vor_und_zurueck()
```