## Bedingte Anweisungen (if ohne else) unter Verwendung von Zufallszahlen mit Turtle

### Aufgabe 1 - Nur auf der rechten Hälfte zeichnen

Wir zoomen hinaus. Die Turtle ist in ihrem sandigen Garten, links davon das Meer, rechts davon die Wüste. Es ist zu heiß und unsere Turtle hat eine Idee. Sich im Sand eingraben und schlafen. Dazu würde sie gerne einen geeigneten Ort in der Wüste suchen. Da sie keine Ahnung hat wohin sie gehen soll, verwendet sie [geoguessr.com](https://www.geoguessr.com/maps), wählt ihr Land aus und erzeugt damit einen zufälligen Ort. Diesen gibt sie in einer Map-App ein und lässt sich dorthin navigieren. Ein Problem ist nur, dass dieser Ort auch im Meer sein kann. Falls das passiert bleibt unsere Turtle demotiviert im Haus stehen und tut nichts. Falls es in die Wüste ist, geht sie dorthin und gräbt sich ein. Bevor sie losgeht drückt diese einen Faden in den Boden um ihr am Haus wiederzufinden.

Wir setzen die Logik der Bedingten Anweisung folgendermaßen um: \* **WENN** die x-Koordinate in der rechten Bildschirmhälfte liegt (Wüste), **DANN** bewege die Turtle dorthin. Danach gräbt sich diese im Sand ein.

**Hinweise:** Verwende dazu folgende Prozeduren: \* pendown(): Die Turtle legt **einen** Faden am Boden ab. Diese malt dadruch Linien wenn diese sich **später** bewegt. \* goto(-100, 200): Die Turtle bewegt sich in einer *geraden Linie* zu der angegebenen *Position*. Die *Position* wird in *x* und *y* *Koordinaten* abegeben. Hier ist die Mitte des Fensters *x = 0* und *y = 0* ist. \* hideturtle(): Die Turtle gräbt sich ein und versteckt sich. \* stamp(): Die Turtle drückt sich auf den Boden und kann somit einen Faden befestigen.

Sowie folgende Funktionen: \* randint(3, 8): Wir ziehen eine zufällige Zahl ohne Kommastellen von z.B. 3 bis 8. Wir wollen hier die Breite und Höhe des Fensters in der sich die Turtle befindet verwenden. \* window\_width(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *breite = window\_width()*. \* window\_height(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *hoehe = window\_height()*.

<img style="" height="240" width="240" src="../images/5.1.1-left.png">  
<img style="" height="240" width="240" src="../images/5.1.1-right.gif">

**Lösung:**

from turtle import \*  
from random import randint  
  
# --- Vorbereitung ---  
shape("turtle")  
# Wir verwenden mit dem Wert 1 eine sichtbare Geschwindigkeit der Turtle.  
speed(1)  
  
# --- Logik ---  
# Wir generierem zufällige Koordinaten innerhalb der Fenstergröße.  
# Das Wort Zufall wird im Englischen das Wort random.   
breite = window\_width()  
hoehe = window\_height()  
  
# Die Bildschrimbreite geht von z.B. 0 bis 100. Für unsre Turtle ist jedoch die Mitte dieser Breite der 0-Punkt.  
# Dadurch ändert sich 0 bis 100 zu -50 bis 50.  
# Um die Turtle zufällig zu navigieren, müssen wir eine Zahl zwischen -50 und 50 ziehen.   
# Das gleiche gilt für die Hoehe.  
# Lege zwei Variable an die die hälfte der bildschirmbreite bzw. der bildschirmhoehe ist.  
halbe\_breite\_mit\_komma = breite / 2   
halbe\_hoehe\_mit\_komma = hoehe / 2   
  
# Die Division ergibt immer eine Kommazahl (float), auch wenn wir 4 / 2 = 2.0 rechnen.  
# Wir müssen deshalb das Ergebnis in eine Zahl ohne Komma (einen Integer) umwandlen.   
halbe\_breite\_ohne\_komma = int(halbe\_breite\_mit\_komma)   
halbe\_hoehe\_ohne\_komma = int(halbe\_hoehe\_mit\_komma)   
  
# Info: Es kann natürlich der Ausdruck breite / 2 direkt in die Funktion int hineingeschrieben werden.  
# halbe\_breite\_ohne\_komma = int(breite/2)  
  
# Info: Oder wir verwenden eine Division, wo eine Zahl ohne Komma rauskommt. Die Ganzzahldivision.  
# Diese ist unter 5 dividiert durch 2 ist 2, mit 1 Rest bekannt und wird mit Python mit // geschrieben.  
# Das 1 Rest wird hier mit // ignorieret.  
# halbe\_breite\_ohne\_komma = breite // 2   
# halbe\_hoehe\_ohne\_komma = hoehe // 2  
  
# Da wir zufällige Zahlen ohne Komma wollen, verwenden wir die Funktion randint.  
# Diese benötigt zwei zahlen, zwischen denen sie zufällig eine Wählt.   
# randint(3, 8) gibt eine Zahl welche 3 sein kann, 8 sein kann und alles dazwischen. Alles dazwischen ist 4, 5, 6 und 7.  
# Der Name ist eine Kombination aus Random und Integer, was Zufall und Zahl ohne Komma bedeutet.  
ziel\_in\_x = randint(-halbe\_breite\_ohne\_komma, halbe\_breite\_ohne\_komma)  
ziel\_in\_y = randint(-halbe\_hoehe\_ohne\_komma, halbe\_hoehe\_ohne\_komma)  
  
# Die Turtle bewegt sich nur zu den Koordinaten, wenn wir uns zwischen 0 und einem Viertel der Breite befinden.  
if ziel\_in\_x > 0:  
 # Hier ist die Einrückung wichtig! Drücke die Tabulator Taste dazu, (links neben dem Q).  
 # Die Turtle drückt den Faden auf den Boden und machen damit einen Abdruck.  
 stamp()   
   
 # Die Turtle holt den Faden aus der Tasche und legt diesen auf den Boden, wenn diese losgeht.  
 pendown()   
   
 # Die Turtle geht in einer geraden linie zu den angegebenen Kooridinaten.  
 goto(ziel\_in\_x, ziel\_in\_y)  
  
 # Am schluss gräbt sich die Turtle ein um im Sand zu schlafen.  
 hideturtle()   
  
# --- Abschluss ---  
# Schließt das Fenster nicht, wenn das Programm beendet ist.  
done()

### Aufgabe 2 - Nur in einem links und rechts abgegrenzten Bereich zeichnen

Unsere Turtle ist immer noch zu heiß. Sie will nun nicht mehr so weit nach rechts gehen. Sie erweitert ihr *demotiviert nichts tun* zusätzlich auf Ziele welche weiter rechts als die hälfte von der Wüste liegen.

Wir setzen die Logik der Bedingten Anweisung folgendermaßen um: \* **WENN** die x-Koordinate in der rechten Bildschirmhälfte liegt (Wüste) und und in der *vorderen Hälfte* der rechten Bildschirmhälfte (Wüste) liegt, **DANN** bewege die Turtle dorthin. Danach gräbt sich diese im Sand ein.

**Hinweise:** Verwende dazu folgende Prozeduren: \* pendown(): Die Turtle legt **einen** Faden am Boden ab. Diese malt dadruch Linien wenn diese sich **später** bewegt. \* goto(-100, 200): Die Turtle bewegt sich in einer *geraden Linie* zu der angegebenen *Position*. Die *Position* wird in *x* und *y* *Koordinaten* abegeben. Hier ist die Mitte des Fensters *x = 0* und *y = 0* ist. \* hideturtle(): Die Turtle gräbt sich ein und versteckt sich. \* stamp(): Die Turtle drückt sich auf den Boden und kann somit einen Faden befestigen.

Sowie folgende Funktionen: \* randint(3, 8): Wir ziehen eine zufällige Zahl ohne Kommastellen von z.B. 3 bis 8. Wir wollen hier die Breite und Höhe des Fensters in der sich die Turtle befindet verwenden. \* window\_width(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *breite = window\_width()*. \* window\_height(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *hoehe = window\_height()*.

<img style="" height="240" width="240" src="../images/5.1.1-left.png">  
<img style="" height="240" width="240" src="../images/5.1.1-right.gif">

**Lösung:**

from turtle import \*  
from random import randint  
  
# --- Vorbereitung ---  
shape("turtle")  
# Wir verwenden mit dem Wert 1 eine sichtbare Geschwindigkeit der Turtle.  
speed(1)  
  
# --- Logik ---  
# Wir generierem zufällige Koordinaten innerhalb der Fenstergröße.  
# Das Wort Zufall wird im Englischen das Wort random.   
breite = window\_width()  
hoehe = window\_height()  
  
# Die Bildschrimbreite geht von z.B. 0 bis 100. Für unsre Turtle ist jedoch die Mitte dieser Breite der 0-Punkt.  
# Dadurch ändert sich 0 bis 100 zu -50 bis 50.  
# Um die Turtle zufällig zu navigieren, müssen wir eine Zahl zwischen -50 und 50 ziehen.   
# Das gleiche gilt für die Hoehe.  
# Lege zwei Variable an die die hälfte der bildschirmbreite bzw. der bildschirmhoehe ist.  
halbe\_breite\_mit\_komma = breite / 2   
halbe\_hoehe\_mit\_komma = hoehe / 2   
  
# Die Division ergibt immer eine Kommazahl (float), auch wenn wir 4 / 2 = 2.0 rechnen.  
# Wir müssen deshalb das Ergebnis in eine Zahl ohne Komma (einen Integer) umwandlen.   
halbe\_breite\_ohne\_komma = int(halbe\_breite\_mit\_komma)   
halbe\_hoehe\_ohne\_komma = int(halbe\_hoehe\_mit\_komma)   
  
# Info: Es kann natürlich der Ausdruck breite / 2 direkt in die Funktion int hineingeschrieben werden.  
# halbe\_breite\_ohne\_komma = int(breite/2)  
  
# Info: Oder wir verwenden eine Division, wo eine Zahl ohne Komma rauskommt. Die Ganzzahldivision.  
# Diese ist unter 5 dividiert durch 2 ist 2, mit 1 Rest bekannt und wird mit Python mit // geschrieben.  
# Das 1 Rest wird hier mit // ignorieret.  
# halbe\_breite\_ohne\_komma = breite // 2   
# halbe\_hoehe\_ohne\_komma = hoehe // 2  
  
# Da wir zufällige Zahlen ohne Komma wollen, verwenden wir die Funktion randint.  
# Diese benötigt zwei zahlen, zwischen denen sie zufällig eine Wählt.   
# randint(3, 8) gibt eine Zahl welche 3 sein kann, 8 sein kann und alles dazwischen. Alles dazwischen ist 4, 5, 6 und 7.  
# Der Name ist eine Kombination aus Random und Integer, was Zufall und Zahl ohne Komma bedeutet.  
ziel\_in\_x = randint(-halbe\_breite\_ohne\_komma, halbe\_breite\_ohne\_komma)  
ziel\_in\_y = randint(-halbe\_hoehe\_ohne\_komma, halbe\_hoehe\_ohne\_komma)  
  
# Die Turtle bewegt sich nur zu den Koordinaten, wenn wir uns zwischen 0 und einem Viertel der Breite befinden.  
if 0 < ziel\_in\_x and ziel\_in\_x <= breite/4:  
 # Hier ist die Einrückung wichtig! Drücke die Tabulator Taste dazu, (links neben dem Q).  
 # Die Turtle drückt den Faden auf den Boden und machen damit einen Abdruck.  
 stamp()   
   
 # Die Turtle holt den Faden aus der Tasche und legt diesen auf den Boden, wenn diese losgeht.  
 pendown()   
   
 # Die Turtle geht in einer geraden linie zu den angegebenen Kooridinaten.  
 goto(ziel\_in\_x, ziel\_in\_y)  
  
 # Am schluss gräbt sich die Turtle ein um im Sand zu schlafen.  
 hideturtle()   
  
# --- Abschluss ---  
# Schließt das Fenster nicht, wenn das Programm beendet ist.  
done()

### Aufgabe 3 - Nur in einem links und rechts oder nach unten abgegrenzten Bereich zeichnen

Unsere Turtle ist immer noch zu heiß. Sie will wie zuvor nicht mehr so weit nach rechts gehen. Zusätzlich ist ihr aufgefallen, wenn sie nach Norden geht, ist es kälter. Ihr macht es dadurch es nichts mehr aus tief in die Wüste hineinzugehen, wenn sie nach norden geht.

Wir setzen die Logik der Bedingten Anweisung folgendermaßen um: \* **WENN** die x-Koordinate in der rechten Bildschirmhälfte liegt (Wüste) und und in der *vorderen Hälfte* der rechten Bildschirmhälfte (Wüste) liegt oder x-Koordinate in der rechten Bildschirmhälfte liegt (Wüste) und die y-Koordinate in der oberen Bildschirmhäfte liegt, **DANN** bewege die Turtle dorthin. Danach gräbt sich diese im Sand ein.

**Hinweise:** Verwende dazu folgende Prozeduren: \* pendown(): Die Turtle legt **einen** Faden am Boden ab. Diese malt dadruch Linien wenn diese sich **später** bewegt. \* goto(-100, 200): Die Turtle bewegt sich in einer *geraden Linie* zu der angegebenen *Position*. Die *Position* wird in *x* und *y* *Koordinaten* abegeben. Hier ist die Mitte des Fensters *x = 0* und *y = 0* ist. \* hideturtle(): Die Turtle gräbt sich ein und versteckt sich. \* stamp(): Die Turtle drückt sich auf den Boden und kann somit einen Faden befestigen.

Sowie folgende Funktionen: \* randint(3, 8): Wir ziehen eine zufällige Zahl ohne Kommastellen von z.B. 3 bis 8. Wir wollen hier die Breite und Höhe des Fensters in der sich die Turtle befindet verwenden. \* window\_width(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *breite = window\_width()*. \* window\_height(): Gibt die breite unseres Fenstes zurück. Merke dir die Antwort mit einer Variable. Das geht so *hoehe = window\_height()*.

<img style="" height="240" width="240" src="../images/5.1.3-left.png">  
<img style="" height="240" width="240" src="../images/5.1.3-bot-right.gif">  
<img style="" height="240" width="240" src="../images/5.1.3-up-right.gif">

**Lösung:**

from turtle import \*  
from random import randint  
  
# --- Vorbereitung ---  
shape("turtle")  
# Wir verwenden mit dem Wert 1 eine sichtbare Geschwindigkeit der Turtle.  
speed(1)  
  
# --- Logik ---  
# Wir generierem zufällige Koordinaten innerhalb der Fenstergröße.  
# Das Wort Zufall wird im Englischen das Wort random.   
breite = window\_width()  
hoehe = window\_height()  
  
# Die Bildschrimbreite geht von z.B. 0 bis 100. Für unsre Turtle ist jedoch die Mitte dieser Breite der 0-Punkt.  
# Dadurch ändert sich 0 bis 100 zu -50 bis 50.  
# Um die Turtle zufällig zu navigieren, müssen wir eine Zahl zwischen -50 und 50 ziehen.   
# Das gleiche gilt für die Hoehe.  
# Lege zwei Variable an die die hälfte der bildschirmbreite bzw. der bildschirmhoehe ist.  
halbe\_breite\_mit\_komma = breite / 2   
halbe\_hoehe\_mit\_komma = hoehe / 2   
  
# Die Division ergibt immer eine Kommazahl (float), auch wenn wir 4 / 2 = 2.0 rechnen.  
# Wir müssen deshalb das Ergebnis in eine Zahl ohne Komma (einen Integer) umwandlen.   
halbe\_breite\_ohne\_komma = int(halbe\_breite\_mit\_komma)   
halbe\_hoehe\_ohne\_komma = int(halbe\_hoehe\_mit\_komma)   
  
# Info: Es kann natürlich der Ausdruck breite / 2 direkt in die Funktion int hineingeschrieben werden.  
# halbe\_breite\_ohne\_komma = int(breite/2)  
  
# Info: Oder wir verwenden eine Division, wo eine Zahl ohne Komma rauskommt. Die Ganzzahldivision.  
# Diese ist unter 5 dividiert durch 2 ist 2, mit 1 Rest bekannt und wird mit Python mit // geschrieben.  
# Das 1 Rest wird hier mit // ignorieret.  
# halbe\_breite\_ohne\_komma = breite // 2   
# halbe\_hoehe\_ohne\_komma = hoehe // 2  
  
# Da wir zufällige Zahlen ohne Komma wollen, verwenden wir die Funktion randint.  
# Diese benötigt zwei zahlen, zwischen denen sie zufällig eine Wählt.   
# randint(3, 8) gibt eine Zahl welche 3 sein kann, 8 sein kann und alles dazwischen. Alles dazwischen ist 4, 5, 6 und 7.  
# Der Name ist eine Kombination aus Random und Integer, was Zufall und Zahl ohne Komma bedeutet.  
ziel\_in\_x = randint(-halbe\_breite\_ohne\_komma, halbe\_breite\_ohne\_komma)  
ziel\_in\_y = randint(-halbe\_hoehe\_ohne\_komma, halbe\_hoehe\_ohne\_komma)  
  
# Die Turtle bewegt sich zu den Koordinaten, wenn wir uns zwischen 0 und einem Viertel der Breite befinden.  
# Die Turtle bewegt sich auch zu den Koordinaten, wenn wir uns größer als 0 in der Breite und größer als 0 in der Höhe befinden.  
if (0 < ziel\_in\_x and ziel\_in\_x <= breite/4) or (ziel\_in\_x > 0 and ziel\_in\_y > 0):  
 # Hier ist die Einrückung wichtig! Drücke die Tabulator Taste dazu, (links neben dem Q).  
 # Die Turtle drückt den Faden auf den Boden und machen damit einen Abdruck.  
 stamp()   
   
 # Die Turtle holt den Faden aus der Tasche und legt diesen auf den Boden, wenn diese losgeht.  
 pendown()   
   
 # Die Turtle geht in einer geraden linie zu den angegebenen Kooridinaten.  
 goto(ziel\_in\_x, ziel\_in\_y)  
  
 # Am schluss gräbt sich die Turtle ein um im Sand zu schlafen.  
 hideturtle()   
  
# --- Abschluss ---  
# Schließt das Fenster nicht, wenn das Programm beendet ist.  
done()

### Aufgabe 3.1 - Variablen statt Werte in der Bedingung verwenden

Da wir schon komplizierte Bedingungen in unserer Bedingten Anweisung haben, wollen wir diese durch Variablen benennen. Das erleichtert uns das Lesen des Codes. Ansonsten ist es hier die gleiche Angabe wie in [Aufgabe 3](#X04e34b90abeb1b47bbd06f800db3244b7bf331c).

Verwende nun Variablen für: \* den logische Ausdruck *Turtle geht in den norden und nach rechts*, \* den logische Ausdruck *Turtle geht nach rechts und nicht zu weit nach rechts* und \* einen logischen Ausdruck die beide vorherigen zusammenfasst. Diese ist unsere Bedingung für die Bedingte Anweisung.

**Lösung:**

from turtle import \*  
from random import randint  
  
# --- Vorbereitung ---  
shape("turtle")  
# Wir verwenden mit dem Wert 1 eine sichtbare Geschwindigkeit der Turtle.  
speed(1)  
  
# --- Logik ---  
# Wir generierem zufällige Koordinaten innerhalb der Fenstergröße.  
# Das Wort Zufall wird im Englischen das Wort random.   
breite = window\_width()  
hoehe = window\_height()  
  
# Die Bildschrimbreite geht von z.B. 0 bis 100. Für unsre Turtle ist jedoch die Mitte dieser Breite der 0-Punkt.  
# Dadurch ändert sich 0 bis 100 zu -50 bis 50.  
# Um die Turtle zufällig zu navigieren, müssen wir eine Zahl zwischen -50 und 50 ziehen.   
# Das gleiche gilt für die Hoehe.  
  
# Die Ganzzahldivision.  
# Diese ist unter 5 dividiert durch 2 ist 2, mit 1 Rest bekannt und wird mit Python mit // geschrieben.  
# Das 1 Rest wird hier mit // ignorieret.  
halbe\_breite\_ohne\_komma = breite // 2   
halbe\_hoehe\_ohne\_komma = hoehe // 2  
  
# Da wir zufällige Zahlen ohne Komma wollen, verwenden wir die Funktion randint.  
# Diese benötigt zwei zahlen, zwischen denen sie zufällig eine Wählt.   
# randint(3, 8) gibt eine Zahl welche 3 sein kann, 8 sein kann und alles dazwischen. Alles dazwischen ist 4, 5, 6 und 7.  
# Der Name ist eine Kombination aus Random und Integer, was Zufall und Zahl ohne Komma bedeutet.  
ziel\_in\_x = randint(-halbe\_breite\_ohne\_komma, halbe\_breite\_ohne\_komma)  
ziel\_in\_y = randint(-halbe\_hoehe\_ohne\_komma, halbe\_hoehe\_ohne\_komma)  
  
# Die Turtle bewegt sich zu den Koordinaten, wenn wir uns zwischen 0 und einem Viertel der Breite befinden.  
# Die Turtle bewegt sich auch zu den Koordinaten, wenn wir uns größer als 0 in der Breite und größer als 0 in der Höhe befinden.  
gehe\_nach\_nicht\_zu\_weit\_in\_die\_wueste = 0 < ziel\_in\_x and ziel\_in\_x <= breite/4  
gehe\_nach\_norden\_und\_weit\_in\_die\_wueste = ziel\_in\_x > 0 and ziel\_in\_y > 0  
  
verhalten\_bei\_hitze = gehe\_nach\_nicht\_zu\_weit\_in\_die\_wueste or gehe\_nach\_norden\_und\_weit\_in\_die\_wueste  
  
if verhalten\_bei\_hitze:  
 # Hier ist die Einrückung wichtig! Drücke die Tabulator Taste dazu, (links neben dem Q).  
 # Die Turtle drückt den Faden auf den Boden und machen damit einen Abdruck.  
 stamp()   
   
 # Die Turtle holt den Faden aus der Tasche und legt diesen auf den Boden, wenn diese losgeht.  
 pendown()   
   
 # Die Turtle geht in einer geraden linie zu den angegebenen Kooridinaten.  
 goto(ziel\_in\_x, ziel\_in\_y)  
  
 # Am schluss gräbt sich die Turtle ein um im Sand zu schlafen.  
 hideturtle()   
  
# --- Abschluss ---  
# Schließt das Fenster nicht, wenn das Programm beendet ist.  
done()