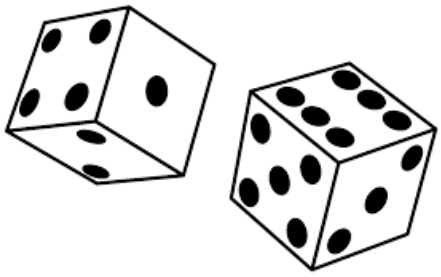
**Lektion 7: micro:bit mit Python** **Anwendung: Wurf zweier Würfel**

In dieser Anwendung wird ein Programm entwickelt, mit dem Daten unter Verwendung des micro:bit gesammelt werden. Während das Programm läuft, beobachtet man, wie ein Punktdiagramm auf dem geteilten Display des TI-nspire wächst.

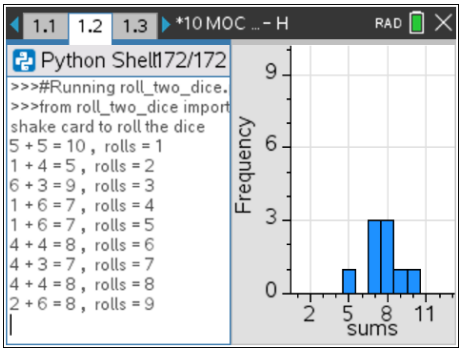
**Lernziele:**

- Schreiben eines Datenerfassungprogrammes für den micro:bit
- Erstellen eines dynamischen Data & Statistic-Diagrammes der gesammelten Daten

1. Diese Anwendung beruht auf den letzten drei micro:bit-Übungen: Schreiben eines Programmes, das eine Geste wie "Shake" (oder einen Tastendruck) verwendet, um einige Daten zu sammeln, speichern der Liste als TI-Nspire-Variable und...

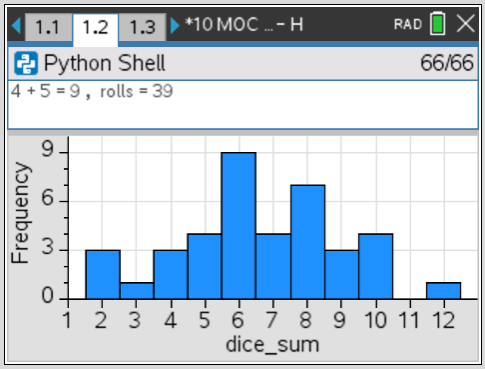


2. … dann die Einrichtung einer TI-nspire-Seite, so dass
   - das Python-Programm auf dem einen Teil der Seite läuft (die Python Shell) und
   - auf der anderen Seite sich ein Data- & Statistic-Plot aufbaut.




**Lehrertipp:** Das oben gezeigte TI-Nspire Split-Screen-Layout kann entweder vertikal oder horizontal sein. Wenn Sie Strg-4 drücken, um zwei Apps auf einer Seite zu gruppieren, ist der Voreinstellung vertikal, wie oben dargestellt. **Um zum horizontalen Layout auf dem Handheld zu wechseln, drücken Sie [doc] > Seitenlayout > Layout auswählen > Layout 3.**

Sie können auch die Trennleiste anpassen, um die Python-Shell-App kleiner und die Data & Statistics-App größer zu machen:




Diese Lektion erzeugt ein Punktdiagramm, kein Histogramm. Anweisungen zum Konvertieren von einem Punktdiagramm in ein Histogramm in der Data & Statistics-App finden Sie weiter unten im Lehrermaterial.

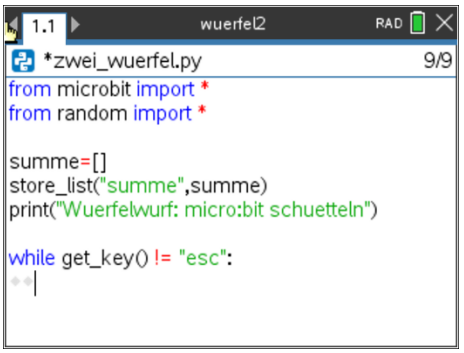
3. Beginnen Sie Ihr micro:bit-Programm mit den üblichen Importen einschließlich des Zufallsmoduls und einer leeren Liste namens

**summe = [ ]**

*Speichern Sie diese Liste sofort* in einer TI-Nspire-Variablen mit demselben Namen), die zunächst also auch leer ist:

**store_list(“summe”, summe)**

**print( )** erteilt eine Anweisung an den Benutzer, bevor die Schleife beginnt. Es wird die "Shake"-Geste verwendet, um die Würfel zu würfeln.




4. Verwenden Sie in der **While**-Schleifen die Geste, um

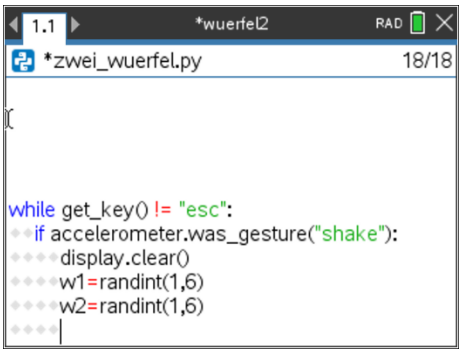
- *zwei Würfel zu werfen* (es werden zwei zufällige ganze Zahlen erzeugt)
- **addieren** Sie sie
- **hängen** Sie die Summe an die **Summenliste** an
- **drucken** Sie die beiden Würfelwerte, ihre Summe und die Wurfnummer (Anzahl der Würfe bisher) auf dem TI-Nspire-Bildschirm. Hinweis: **len(summe)** ist die Anzahl der Listenelemente und damit die Wurfnummer
- **anzeige** beider Würfel-Werte auf dem micro:bit
- **speichern** der **Liste** in einer TI-Nspire-Variablen

**Lehrertipp:** Die Schüler sollten über genügend Wissen aus den Übungen 1, 2 und 3 verfügen, um dieses Programm zu entwickeln. Wenn sie mit Menüs zu kämpfen haben, ermutigen Sie sie, in den vorherigen Übungen nachzuschauen.

5. So sieht der Würfelwurf aus (alternativ kann auch ein Tastendruck verwendet werden):

```
    if accelerometer.was_gesture("shake"):
        display.clear()
        w1 = randint(1,6)
        w2 = randint(1,6)
```

*Achten Sie auf die Einrückungen.*




6. Bilden Sie die Summe und fügen (**append)** Sie die Summe als beues letztes Element der Liste der Summen hinzu

```
s = w1 + w2
summe.append(s)
```

7. Zeigen Sie die beiden Würfe auf dem micro:bit-Display an. Denken Sie daran, dass die beiden Würfe möglicherweise den gleichen Wert haben, daher muss durch eine Pause sichergestellt werden, dass beide tatsächlich angezeigt werden.:


```
display.clear()
display.show(w1)
sleep(250)
display.clear()
display.show(w2)
sleep(250)
```

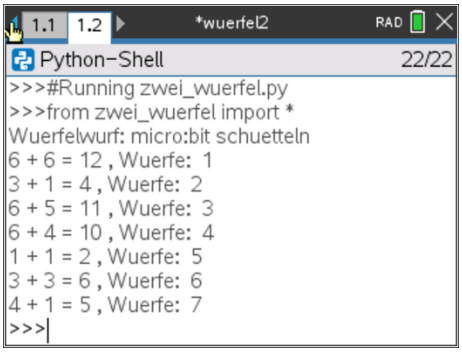
*Eine andere Verzögerung der sleep()-Anweisung ist* u.U. sinnvoll. *Versuchen Sie jetzt, das Programm auszuführen und schütteln Sie das micro:bit. Auf dem micro:bit sollten zwei Zahlen angezeigt werden.*

**Lehrertipp:** Eine weitere **sleep( )** – Anweisung in der Schleife kann hilfreich sein, damit micro:bit die Ausführung der Geste besser überwachen kann**.**

**8.** Mit einer einzigen **print()** - Anweisung können die einzelnen Würfelwerte, ihre Summe und die Wurfnummer auf dem TI-Schirm ausgegeben werden:

**print (w1, "+", w2,"=",s,", ","Wuerfe: ", len(summe))**

Man erhält die nebenstehende Abbildung.




9. Nun muss noch die Python-Liste **summe** in eine TI-nspire-Liste gleichen Namens abgespeichert werden:

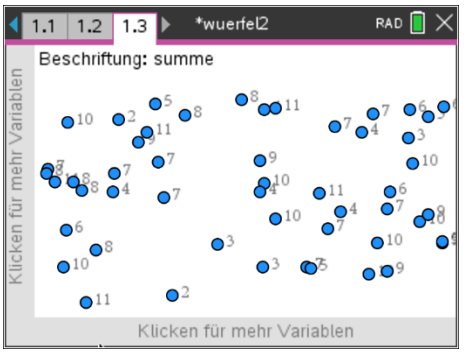

```
    store_list("summe", summe)
```

*Jedes Mal, wenn gewürfelt wird, werden diese Listen aktualisiert, da die Anweisung am Ende des* ***while****-Blockes steht.*

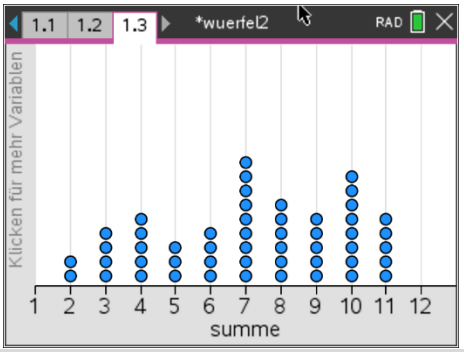
**Lehrertipp:** Beachten Sie das Fehlen einer "Zähler"-Variablen in der obigen Druckanweisung. Es wird nicht benötigt, da dafür die Länge der Listen verwendet wird**.**

10. Wenn Sie überzeugt sind, dass Ihr Programm ordnungsgemäß funktioniert, können Sie Ihr Python-Programm mit den TI-Nspire-Plotfunktionen verbinden. Führen Sie Ihr Programm aus und generieren Sie etwa 50 Würfe. Drücken Sie **[esc],** um das Programm zu beenden.

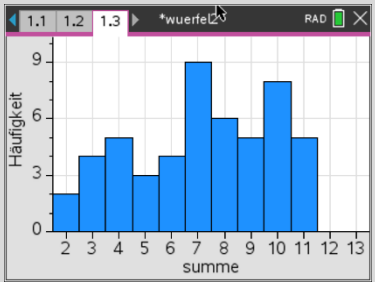
    In der Python Shell (also beim Prompt **>>>**) kann man durch Drücken von **[ctrl] [doc]** oder **[ctrl] [I]** eine Seite einfügen, in diesem Fall **Data & Statistics**. Man sollte ein Bild wie nebenstehend sehen mit über den Schirm verteilten Datenpunkten..



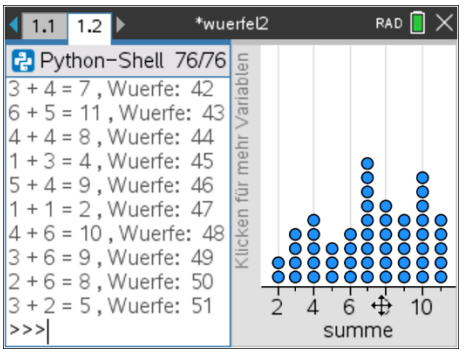

11. Klicken Sie unten auf dem Bildschirm auf die Meldung '*Klicken Sie, um eine Variable hinzuzufügen*' und wählen Sie die Listenvariable **summe** aus. Ihre verstreuten Datenpunkte sind nun entlang der x-Achse entsprechend ihrem Wert organisiert und das Fenster ist für die Daten entsprechend zugeschnitten. Dies ist ein **Punktdiagramm**.



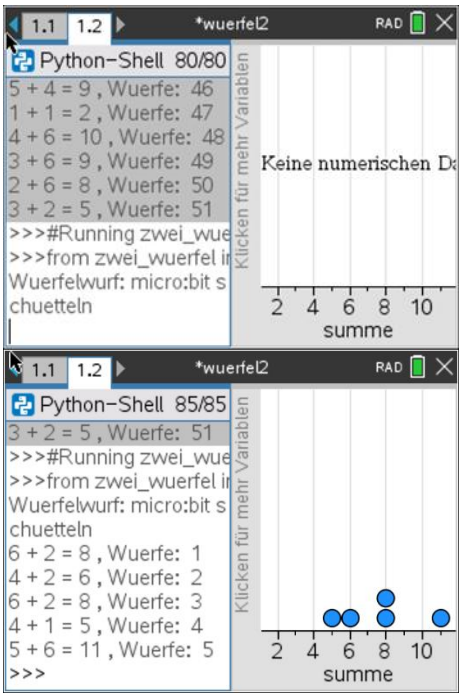

**Lehrertipp:** *Man kann das Punktdiagramm in ein Histogramm verwandeln. Dazu muss man* **[menu] > Plot Type > Histogramm** drücken. Sie sollten aber auch die Säulen-Ausrichtung auf 0,5 anpassen, damit die Balken über ihre x-Achsenwerte zentriert sind. *Dazu* **[menu] > Plot Eigenschaften > Histogramm Eigenschaften > Säuleneinstellungen: gleiche Säulenbreite** *und* **Ausrichtung 0.5***.*




12. Mit (**[ctrl] [nach links]**) kommt man wieder zur Python Shell. Drückt man nun **[ctrl] [4]** , so erhält man einen geteilten Schirm mit der Python Shell links und dem Data&Statistics – Bild rechts.



13. Die Shell wurde "neu initialisiert", sodass das Programm durch Drücken von **[ctrl] [R]** nicht erneut ausgeführt wird. Gehen Sie zurück zum Python-Editor und drücken Sie **[ctrl] [R],** um das Programm auszuführen. Es wird in der Shell mit halbem Bildschirm ausgeführt, wie hier gezeigt. Sie sehen zunächst "Keine numerischen Daten" auf der rechten Seite, da das Programm mit einer leeren Liste startet.




Wenn Sie nun Daten sammeln (schütteln Sie das micro:bit, um die Würfel zu werfen), werden Ihre **Summenwerte** als Punkte in der Data & Statistics-App auf der rechten Seite angezeigt.
Durch Drücken von **[esc]** wird das Programm beendet und Sie können viele andere 1-Variablen-Datenanalysen in der TI-Nspire-Umgebung durchführen.
Wenn Sie **[Strg] [R]** erneut (jetzt in der Python-Shell) drücken, wird das Programm auch erneut ausgeführt.

*Tipp: Um die Shell zu Beginn jedes Laufs zu löschen, fügen Sie zu Beginn Ihres Programms die Anweisung hinzu:*

**clear_history()**

*gefunden auf* **[menu] > Weitere Module > BBC micro:bit > Commands**

Viel Spaß und denken Sie daran, Ihr Dokument zu speichern!

**Lehrertipp:** Das TI-Nspire Split-Screen-Layout kann entweder vertikal oder horizontal sein. Wenn Sie Strg-4 drücken, um zwei Apps auf einer Seite zusammenzuführen, ist die Standardeinstellung vertikal. Um auf dem Handheld zum horizontalen Layout zu wechseln, drücken Sie [doc] > Seitenlayout > Wählen Sie Layout > Layout 3. Sie können auch die Trennleiste anpassen, um die Python-Shell-App zu verkleinern:

**Das vollständige Programm:**

*zwei_wuerfel.py 10/21

```
from microbit import *
from random import *

summe=[]
store_list("summe",summe)
print("Wuerfelwurf: micro:bit schuetteln")
while get_key() != "esc":
    if accelerometer.was_gesture("shake"):
        display.clear()
        w1=randint(1,6)
        w2=randint(1,6)
        s=w1+w2
        summe.append(s)
        display.clear()
        display.show(w1)
        sleep(250)
        display.clear()
        display.show(w2)
        sleep(250)
        print(w1,"+",w2,"=",s,",","Wuerfe: ",len(summe))
```

**Optionale Erweiterung: Anzeige der sechs Würfelflächen**

**Es ist einfach, Bilder auf dem Display des micro:bit zu erzeugen, da man jede der 5x5 LEDs in ihrer Helligkeit von 0 bis 9 einzeln ansteuern kann.**
**Der nachfolgende Code bildet die 6 Seitenflächen des Würfels ab. Man beachte die Großschreibung des „I“ in Image. Zunächst sollte man die erste Seite des Würfels entwerfen, also eins=Image(…, dann lassen sich die übrigen 5 Seiten mittels copy/paste/edit einfach herstellen. Man benötigt noch ein Element null (das erste Listenelement mit der Nummer 0), da der Listenindex stets mit 0 beginnt.**
**Python interpretiert zwei Zeichenfolgen, die in separaten Zeilen ohne Trennzeichen (wie ein Komma) geschrieben wurden, als eine einzelne Zeichenfolge, also**

**“aaa”**
**“bbb”**

**ist dasselbe wie**

**“aaabbb”**

**Als Zeilentrenner wird im folgenden Code der Doppelpunkt verwendet**

```
##############################################################
from microbit import *
from random import *
null=Image(
"00000:"
"00000:"
"00000:"
"00000:"
"00000")
```

```
eins=Image(
"00000:"
"00000:"
"00900:"
"00000:"
"00000")

zwei=Image(
"00000:"
"09000:"
"00000:"
"00090:"
"00000")

drei=Image(
"00000:"
"09000:"
"00900:"
"00090:"
"00000")

vier=Image(
"00000:"
"09090:"
"00000:"
"09090:"
"00000")

fuenf=Image(
"00000:"
"09090:"
"00900:"
"09090:"
"00000")

sechs=Image(
"00000:"
"09090:"
"09090:"
"09090:"
"00000")
```

```
wuerfel=[null, eins, zwei, drei, vier, fuenf, sechs]
#   index:   0     1       2       3     4       5        6
print("running...")
while get_key() != "esc":
  if button_a.is_pressed():
    w=randint(1,6)
    display.show(wuerfel[w])  # Darstellung einer Wuerfelseite
```