



#### Unit 6: micro:bit with Python

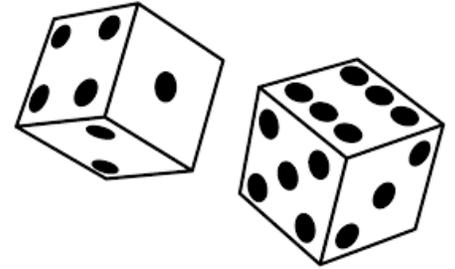
#### Application: Toss the dice

In this Application, you will write a program to collect data using the micro:bit and run the program while observing a dot plot grow on a split page of the TI-Nspire.

#### Objectives:

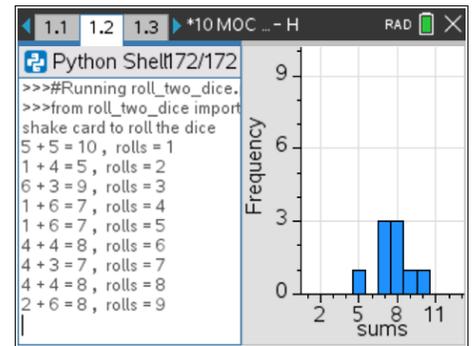
- Write a micro:bit data collection program
- Create a dynamic Data & Statistics plot of the collected data

1. This Application project is a compilation of all the micro:bit skills you learned in the past three Skill Builders: write a program that uses a gesture, like 'shake' (or a button press) to collect some data, store the list as a TI-Nspire variable and...



2. ... then set up a TI-Nspire page that

- runs the python program on one side of the screen (the python Shell) and
- displays a dot plot (or histogram) of the collected data *as you are running the program* (using a Data & Statistics app).



3. Begin your micro:bit program with the usual imports including the **random** module and start with an empty list called **sums**:

```
sums = []
```

Immediately store this list to a TI-Nspire variable (using the same name).

```
store_list("sums", sums)
```

so that the TI-Nspire list is cleared as well.

**print()** some instructions to the user before the loop begins. We are going to use the 'shake' gesture to roll the dice.

```
*roll_two_dice.py 7/30
from random import *
from microbit import *

sums=[]
store_list("sums",sums)
print("shake card to roll the dice")

while get_key() != "esc":
```

4. In the **while** loop body, use the gesture to

- **toss** two dice (generate two random integers)
- **add** them together
- **append** the sum to the **sums** list
- **print** the two dice values, their sum and the roll number on the TI-Nspire screen. Hint: **len(sums)** is the roll number.
- **display** both die values on the micro:bit
- **store** the **list** to a TI-Nspire variable

Try it now.

```
*roll_two_dice.py 7/30
from random import *
from microbit import *

sums=[]
store_list("sums",sums)
print("shake card to roll the dice")

while get_key() != "esc":
```



# 10 Minutes of Code - Python

## MICRO:BIT USING TI-NSPIRE CX II

## UNIT 6: APPLICATION

## STUDENT ACTIVITY

5. To toss the dice use a gesture or button press:

```

◆◆if accelerometer.was_gesture("shake"):
◆◆◆display.clear()
◆◆◆r1 = randint(1,6)
◆◆◆r2 = randint(1,6)

```

*Again, note the indentations.*

6. Add them together and **.append** the **sum** to the **sums** list:

```

sum = r1 + r2
sums.append(sum)

```

7. Display the two dice on the micro:bit display. Remember that the two dice might have the same value so we want to make sure that both will actually appear:

```

display.clear()
display.show(r1)
sleep(250)
display.clear()
display.show(r2)
sleep(250)

```

*You might prefer a longer delay in the **sleep( )** commands.*

*If you have entered the code properly and in the right sequence, try running the program now and shake the micro:bit. You should see two numbers displayed on the micro:bit.*

```

1.1 1.2 1.3 *mb APP RAD
*roll_two_dice.py 13/41
...
...
...
...
if accelerometer.was_gesture("shake"):
display.clear()
r1 = randint(1,6)
r2 = randint(1,6)
...
...
...

```

```

1.1 1.2 1.3 *mb APP RAD
*roll_two_dice.py 37/41
...
...
...
...
sum = r1 + r2
sums.append(sum)
...
...
...

```

```

1.1 1.2 1.3 *mb APP RAD
*roll_two_dice.py 12/28
...
...
...
...
display.clear()
display.show(r1)
sleep(250)
display.clear()
display.show(r2)
sleep(250)
...
...
...

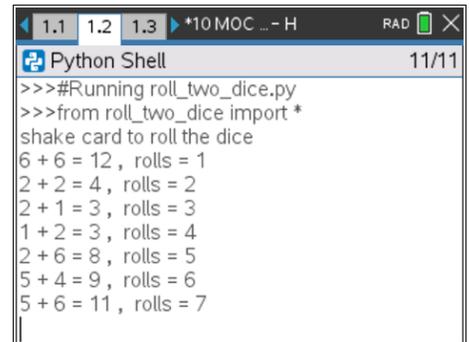
```

8. Add code to print the dice, sum and rolls on the TI-Nspire screen. We can use a single **print()** statement like this:

```
◆◆◆◆ print (r1, "+", r2,"=",sum," ", "rolls =", len(sums))
```

which results in the lines shown in this image.

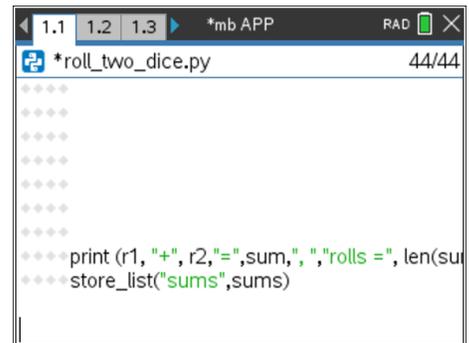
*Be careful about the punctuation!*



9. Store the python list **sums** to a TI-Nspire list of the same name:

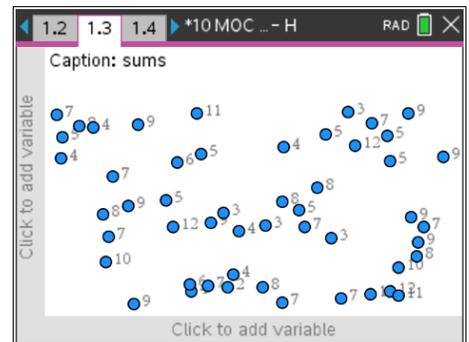
```
◆◆◆◆ store_list("sums", sums)
```

*This store\_list() statement is deep inside the while and if blocks so that the TI-Nspire list is updated every time a new pair of dice is generated.*

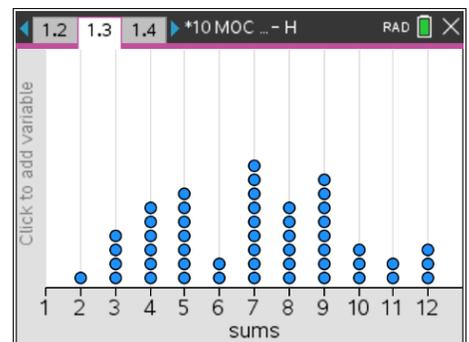


10. When you are satisfied that your program is working properly you are ready to connect your python program to the TI-Nspire plotting capabilities. Run your program and generate about 50 rolls. Press **[esc]** to end the program.

In the python Shell (at the command prompt >>>) press **[ctrl] [doc]** or **[ctrl] [I]** to insert a page. Select the **Data and Statistics** app. You should see a screen similar to the one on the right. Your sums data is scattered around the screen.



11. Click on the 'Click to add variable' message on the bottom of the screen and select the list variable **sums**. Your scattered data points are now organized along the x-axis according to their value and the window is suited to the data. This is a **Dot Plot**.





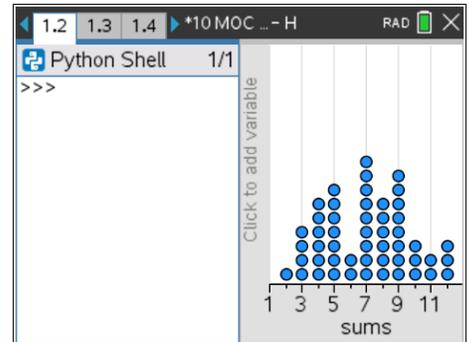
# 10 Minutes of Code - Python

## MICRO:BIT USING TI-NSPIRE CX II

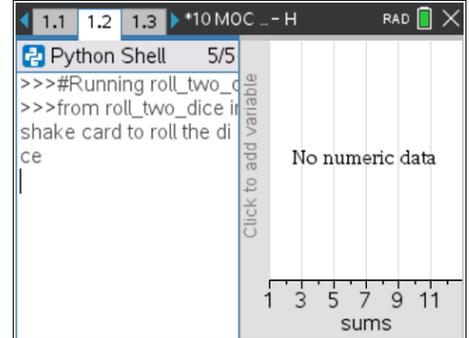
## UNIT 6: APPLICATION

## STUDENT ACTIVITY

12. Go back one page to the python Shell app (**[ctrl] [leftarrow]**) and press **[ctrl] [4]** to 'group' this app with the Data and Statistics app, creating a split-screen page with your python Shell on the left and your Data & Statistics app on the right as shown here.



13. The Shell has been 're-initialized' so pressing **[ctrl] [R]** will not re-run the program. Go back to the python Editor and press **[ctrl] [R]** to run the program. It runs in the half-screen Shell as shown here. You see 'No numeric data' on the right because the program stores an empty list right away.



As you collect the data (shake the micro:bit to roll the dice) your **sums** values appear as dots in the Data & Statistics app on the right.

Pressing **[esc]** will end the program and you can do a lot of other 1-variable data analysis in the TI-Nspire environment.

Pressing **[ctrl] [R]** again now (in the python Shell) does re-run the program.

*Tip: to clear the Shell at the start of each run add the statement:*

**clear\_history()**

*found on [menu] > More Modules > BBC micro:bit > Commands at the beginning of your program.*

Enjoy, and remember to save your document!