

Unit 5: The TI Modules

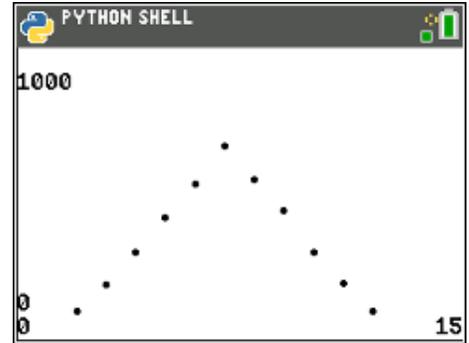
Skill Builder 1: The Plot Thickens

In this lesson, you will create a scatter plot of the dice totals from **Unit 4, Application** using the **tiplotlib** module.

Objectives:

- Introduce **tiplotlib**
- Make a scatter plot
- Adjust the window

In the **Unit 4, Application** you made a simulation of tossing two dice and logging the experiment's totals in a list. Here you will continue with that program and see how easy it is to make a scatter plot of that data using Python.



1. Rather than starting from scratch, load the dice project from the **Unit 4, Application** — we called it DICE.

You can make a copy of your program in the FILE MANAGER by pointing to your DICE program name and using:

<Manage> Replicate Program...

Type a new name when prompted. We'll use **DICEB**.



2. Your DICE program should resemble the image shown here. After the first **for** loop, there is a list called **totals** that contains 11 values representing the number of times each total from 2 to 12 occurred in your trials.

It looks similar to this for 100 trials:

totals = [1, 6, 5, 9, 15, 13, 19, 13, 9, 5, 5]

Your numbers will vary.

The second for loop calculates and then prints the experimental probabilities. We will not be concerned with that loop in this project.

```

from random import *
totals=[0]*11
trials=int(input("# of trials?"))
for i in range(trials):
    die1=randint(1,6)
    die2=randint(1,6)
    sum=die1+die2
    totals[sum-2]=totals[sum-2]+1
print(totals)

for i in range(11):
    totals[i]=round(totals[i]/trials,2)
print(totals)
    
```



- To create a graphical plot of the data that the program generated, you need to import another custom TI module. At the top of your program, below "from random...", add the following import statement:

import ti_plotlib as plt

You can get this entire statement from <Fns...> Modul ti_plotlib.

Note: This form of **import** requires the prefix **plt.** in front of all of the **ti_plotlib** functions, and that will be provided when you select a function from the menu.

- Scroll to the bottom of the program (below **print(totals)**).

To make a scatter plot we need *two* lists, an *xlist* and a *ylist*. **totals** is going to be the *ylist*. For the *xlist*, we use the 11 possible sum values:

sums = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

There are other, clever ways of making this list, but just typing it in is fast and simple.

- Now we can set up and display the scatter plot of (sums, totals).

Setup statements are taken from <Fns...> Modul ti_plotlib Setup.

The order of the functions on the menu illustrates the order in which they should appear in your code, and all Draw functions should follow the setup functions.

The setup we'll use here are the statements:

- plt.cls()** to clear the Shell screen for plotting
- plt.window(, , ,)** to establish the viewing window

The window settings depend on the data. Use -5,15 for the x-axis and 10,1000 for the y-axis (we plan on tossing a lot of dice).

- plt.axes()**

The choices for the "mode" appear on a sub-menu. Select "on" to see the axes *and* window settings.

- To make the scatter plot, from <Fns...> Modul ti_plotlib > Draw menu select:

plt.scatter(xlist, ylist, "mark")

- For *mark*, choose one from the four that are offered on the sub-menu; we chose the circle
- For *xlist*, type **sums**
- For *ylist*, type **totals**

```
EDITOR: DICEB
PROGRAM LINE 0004

from random import *
import ti_plotlib as plt

totals=[0]*11
trials=int(input("# of trials?"))

for i in range(trials):
    die1=randint(1,6)
    die2=randint(1,6)
    sum=die1+die2
```

```
EDITOR: DICEB
PROGRAM LINE 0015

for i in range(trials):
    die1=randint(1,6)
    die2=randint(1,6)
    sum=die1+die2
    totals[sum-2]=totals[sum-2]+1
print(totals)

sums=[2,3,4,5,6,7,8,9,10,11,12]
-
```

```
EDITOR: DICEB
PROGRAM LINE 0013

die2=randint(1,6)
sum=die1+die2
totals[sum-2]=totals[sum-2]+1
print(totals)

sums=[2,3,4,5,6,7,8,9,10,11,12]
plt.cls()
plt.window(-5,15,10,1000)
plt.axes("on")
```

TI-84 PLUS CE PYTHON

STUDENT ACTIVITY

and from <Fns...> Modul ti_plotlib setup menu choose:

plt.show_plot() to pause the display until [clear] is pressed.

```

EDITOR: DICEB
PROGRAM LINE 0020
** totals[sum-2]=totals[sum-2]+1
print(totals)

sums=[2,3,4,5,6,7,8,9,10,11,12]
plt.cls()
plt.window(-5,15,10,1000)
plt.axes("on")
plt.scatter(sums,totals,"o")
plt.show_plot()
    
```

7. Run the program. Enter 1000 for the number of trials.

Does your plot look something like this? It may not match exactly since the dice are random.

Press [clear] to exit the plot screen and return to the Python Shell

Try again with a larger number of trials.

Consider adjusting your **window** values.

Reminder: To “break” a running program press the [on] key.

8. You can customize the window for the number of trials:

In the **plt.window()** function change *y*max to **1.1*max(totals)**.

This image shows 50000 tosses with this change as the program automatically adjusts the ymax value.

Note that we *only* added **five** new statements to the program to make the plot!

