Unit 1: Getting Started with the TI-Innovator™ Hub

Skill Builder 3: Input and Sound

In this lesson, you will learn about two ways to make sound on the TI-Innovator Hub and use variables to store and retrieve values.

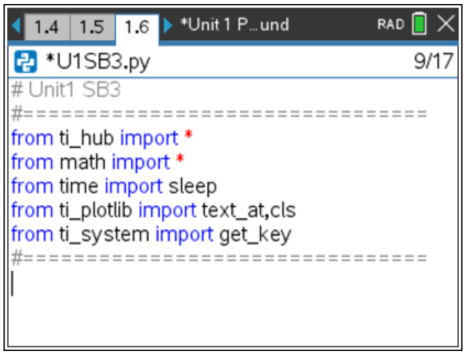
Objectives:

- Produce a sound on the TI-Innovator Hub
- Use variables to store entered values
- Use the **input()** function and the **int()** function
- Play musical notes

The TI-Innovator Hub houses a small speaker, called ‘SOUND’. The speaker is on the bottom of the hub. There is no amplifier, so the volume is very low. It is designed that way so that your classroom does not get too noisy.

1. Start a new Python project using the Hub Project template.

   You can either start a new document (**home > New**) or add a page to your current document (**ctrl+doc**). Select **Add Python > New….** Be sure to select the **Hub Project** template.



```
# Unit1 SB3
#================================
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================
```

Teacher Tip: Python is case sensitive: x and X represent two different things. All keywords are lowercase. Using a capital letter in a keyword like ‘If’ results in an error.

2. You will write two **input** statements to get values to create a sound. These values will be stored in two variables, **f** and **t**. You will write two statements like this:

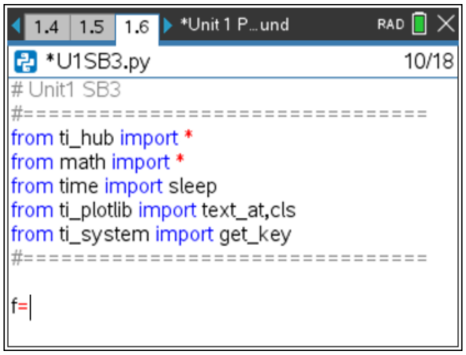
f = input(“Frequency? “)
t = input(“Time? “)

We will do this step-by-step.

3. Type the letter f and the = sign on a blank line below the bottom comment line **#=============**.

f =

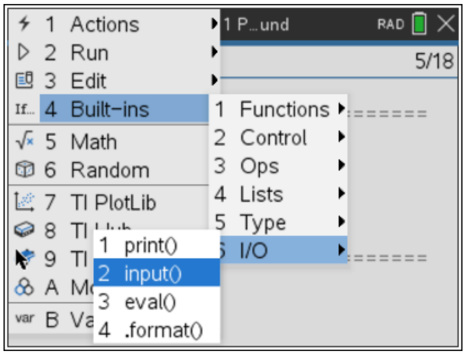
Leave your cursor to the right of the = sign for the next step.



```
# Unit1 SB3
#================================
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================

f=
```

4. Now use the **input()** function from **menu > Built-ins > I/O**.

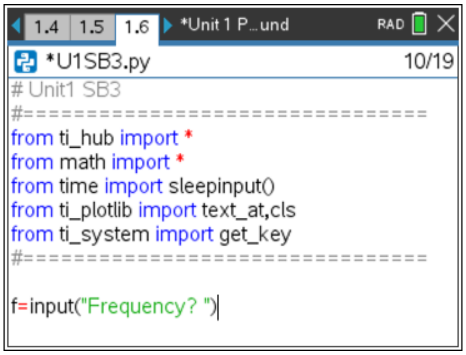


Teacher Tip: About **input()**:
The input function is a built-in Python function that allows a user to enter information to the program while it is running. However, the input function always returns a string value, not a number. The initial run of this program will generate a runtime error that will be fixed later in the lesson.

5. Selecting **input()** from the menu pastes the word **input()** into your program after the = sign. Inside the parentheses, type the "prompt" for this input statement.

   On the handheld, the quotation mark is **ctrl+x** (the multiplication key). For a capital letter use the shift key. Locate the question mark (?) on the punctuation key next to the letter 'g' on the keypad. Be sure to use the closing quote, as well.

   On the next line, build another input statement for the variable **t**.



```
# Unit1 SB3
#================================
from ti_hub import *
from math import *
from time import sleepinput()
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================

f=input("Frequency? ")
```

6. Below the two input statements, get the **sound.tone()** function from

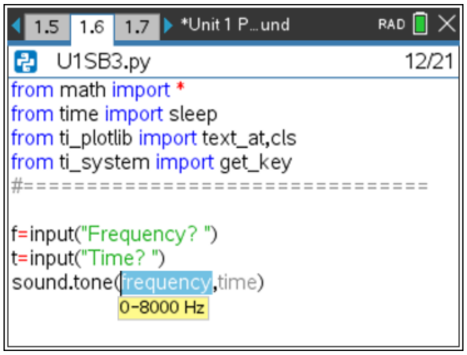
   menu > TI Hub > Hub Built-in Devices > Sound Output.

   You will see the statement

   sound.tone(frequency, time)

   added to your program.

   In place of frequency, type the variable **f**. In place of time, type the variable **t**.

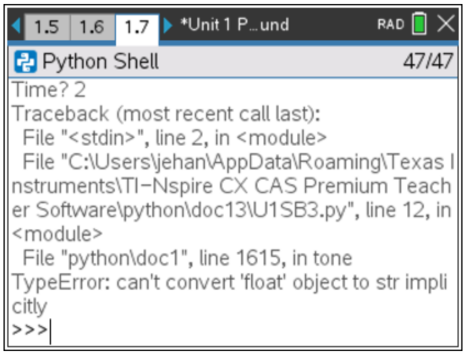


```
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================

f=input("Frequency? ")
t=input("Time? ")
sound.tone(frequency,time)
```

7. Press **ctrl+R** to run the program. At the prompts, enter 440 for Frequency and 2 for Time.

   And then… oops! A 'runtime error'! This error occurs because the input function gives a value that the sound function cannot use; it is a string ("440" and "2") rather than the numeric values 440 and 2. You need to convert each string value to a numeric value using either **float()** (for a decimal number) or **int()** (for an integer).



```
Time? 2
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
  File "C:\Users\jehan\AppData\Roaming\Texas I
nstruments\TI−Nspire CX CAS Premium Teach
er Software\python\doc13\U1SB3.py", line 12, in
<module>
  File "python\doc1", line 1615, in tone
TypeError: can't convert 'float' object to str impli
citly
>>>
```

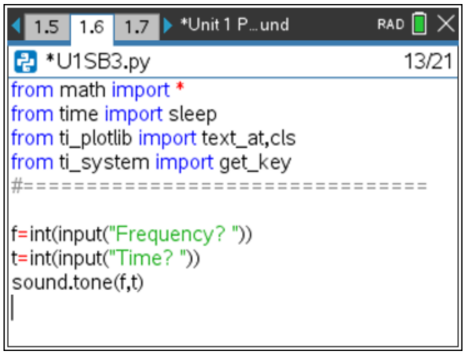
8. Back in your program, add the **int()** function in front of each input function. You can type it in or find the **int()** function on

   Menu > Built-ins > Type.

   f = int(input("Frequency? "))
   t = int(input("Time? "))

   Notice the special placement of all the parentheses! You will have to move one parenthesis around. Notice that there are two right parentheses at the end of each statement.

   Run the program again and enter 440 and 2, you will hear a tone of 440 Hz for 2 seconds from the speaker on the bottom of the TI-Innovator Hub. Press **ctrl+R** to rerun the program in the Shell to try other frequencies (between 0 and 8000 Hz). Can you hear them all?



```
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================

f=int(input("Frequency? "))
t=int(input("Time? "))
sound.tone(f,t)
```

9. Play **Notes**

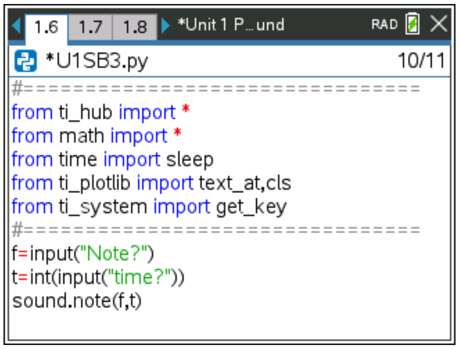
The sound device can also play musical notes using their names:

sound.note(“A4”,2)

plays the note A in the 4th octave.

Change the first input function to request a note rather than a frequency. Since the note is in quotes (a string), you do not need the **int()** function.

Change the sound command from tone to note by retyping it or by using the menu. When you run the program, enter a note like **A4** (without quotes).



```
#================================
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#================================
f=input("Note?")
t=int(input("time?"))
sound.note(f,t)
```

Teacher Tip: Low and high frequencies are inaudible. But, a very low frequency such as 3 or 4 will cause the speaker to ‘click’ that many times per second and you can hear those clicks. Slowly increase the frequency and you will hear the clicks gradually change into tones.

In the **.note()** function, use the letters F and S for flats and sharps such as BF3, DS5, etc.