Unit 3: Conditions, if and while | **Skill Builder 3: Press a key**

In this lesson you will use a function that looks for a *keypress* to terminate a loop. This feature is unique to the TI-Nspire so you will need to import one (or more) special modules.

Objectives:

- Use **get_key()** to end a loop
- Examine powers of 2 and powers of 1/2.
- Use **sleep(n)** to pause code execution for n seconds

Now that you've experienced the **while** loop, let's visit a special and powerful feature of the TI-Nspire Python system: **get_key()**. We will write a program that will display powers of 2 (2, 4, 8, 16, 32, 64, …) continuously and will end only when the **esc** key is pressed.

Teacher Tip: get_key() is a function included in the ti_system module. It is used a great deal in working with drawing programs (a later unit) and TI-Innovator™ Hub programs (in another course).
from ti_system import get_key is needed to be able to use the feature in any program. There are many other functions in that module. Check the online documentation.
get_key() or **get_key(0)** checks for a keypress *without stopping the program*.
get_key(1) will pause the program and wait for the keypress.

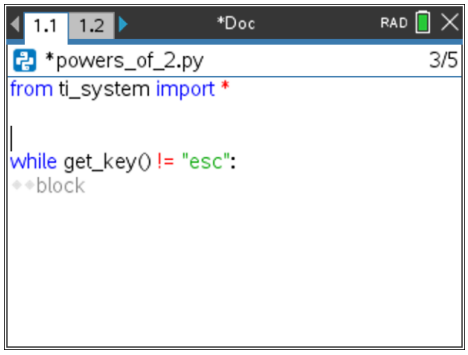
1. Begin a new Python 'blank program' and name it '**powers_of_2**'.
2. Use **menu > More Modules > TI System** to get the statement

   from ti_system import *

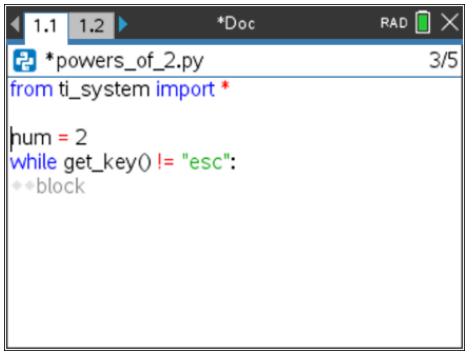
   located at the top of the list.

   On the same **menu**, select the special statement:

   while get_key() != "esc":
   block

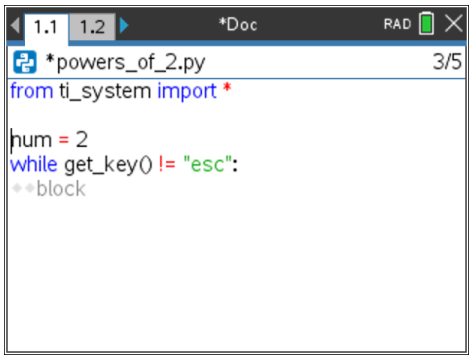


3. Now introduce a variable **num = 2** just *before* the **while** statement.



4. In the loop block, **print** the number and then make an **assignment statement** that will multiply it by 2. Be sure your statements are indented.

   Try it yourself before checking the code in the next step…

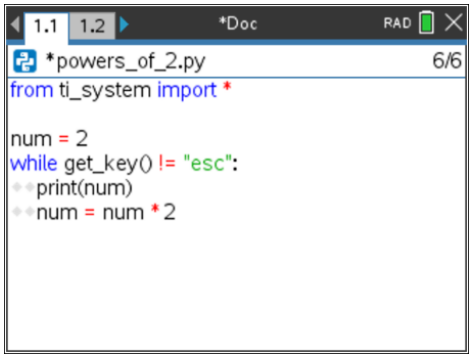


Teacher Tip: Use **num=num*2** or **num*=2**

5. Here you go…

   ```
   while get_key() != “esc”:
       print(num)
       num = num * 2
   ```

   Run the program and press **esc** to stop it.

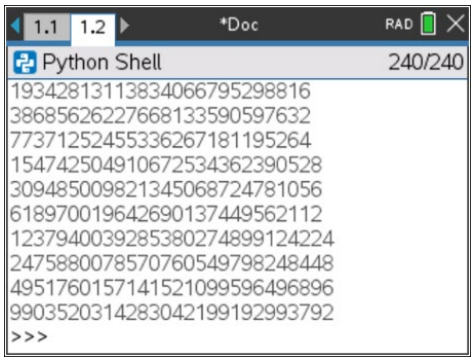


6. The numbers grow really fast!

   Two interesting features of Python:

   It's *really* fast and
   there's *no upper limit* to the integers.
   However, you *are* limited by the capacity of the computer's memory.

   Your next task is to slow things down so that you can read the numbers.



7. You need a function called **sleep()** that is found in the Python **time** module.

   At the top of your program add the statement

   from time import sleep

   located on the top of **menu > More Modules > Time**.

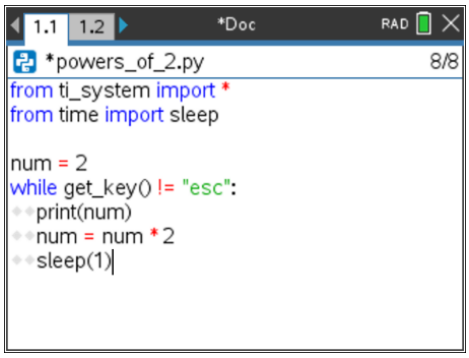
8. Below the statement **num = num * 2** add the statement

   sleep(1)

   inside the **while** block (indented like the other two statements above it).

   Run the program again. This time, after each number is displayed, the computer sleeps (waits) for one second before proceeding to the next step in the loop. Change the sleep number to speed things up a bit. Again, press **esc** to end the program. Try other sleep() values to vary the speed of the display.

   At the end of the program, print “Done”.



9. Change each **2** in the program to **0.5**.

   What happens?

   Try other numbers.

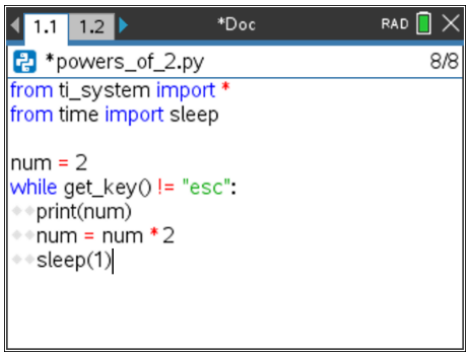
   When do the numbers grow and when do they shrink?

   How about negative numbers?

   Would it make sense to try 0 or 1 instead of 2?

   Now try this: Change **num = num * 2** to **num *= 2**

   This shortcut operation does the same thing and works for many other mathematical operators.



Teacher Tip: **get_key()** and **sleep()** are additional functions that make Python programming very interesting. **time** is a standard Python module that has other time-related functions. **ti_system** is a Texas Instruments module designed to work only with the TI-Nspire CX II Operating System.