14.6. Infinite Loops

Although the textbook has a time limit which prevents an active code window from running indefinitely, you'll still have to wait a little while if your program has an ininite loop. If you accidentally make one outside of the textbook, you won't have that same protection.

So how can you recognize when you are in danger of making an infinite loop?

First off, if the variable that you are using to determine if the while loop should continue is never reset inside the while loop, then your code will have an infinite loop. (Unless of course you use `break` to break out of the loop.)

Additionally, if the while condition is `while True:` and there is no break, then that is another case of an infinite loop!

```python
while True:
    print("Will this stop?")

print("We have escaped.")
```

Another case where an infinite loop is likely to occur is when you have reassiged the value of the variable used in the while statement in a way that prevents the loop from completing. This is an example below (if it takes too long, try reloading the page and stepping through this example in codelens):

Save & Run   5/14/2021, 3:42:31 PM - 2 of 2   Show in CodeLens

```python
1 b = 15
2
3 while b < 60:
4     #b = 5
5     print("Bugs")
6     b = b + 7
7
```

```
Bugs
Bugs
Bugs
Bugs
Bugs
Bugs
Bugs
```

Activity: 1 -- ActiveCode (ac14_11_1)

Notice how in this case, b is initally set to 15 outside of the while loop, and then reassigned the value of 5 inside, on line 4. By the time 7 has been added to b on line 6, we then have to check if b is less than 60. Because it isn't we again run line 4, and set the value of b to 5 again. There is no way to break out of this loop.

Sometimes programs can take a while to run, so how can you determine if your code is just talking a while or if it is stuck inside an infinite loop? Print statements are for people, so take advantage of them! You can add print statements to keep track of how your variables are changing as the program processes the instructions given to them. Below is an example of an infinite loop. Try adding print statments to see where it's coming from. When you're done, check out the answer to see what our solution was.

1. Question   Answer

Save & Run   Original - 1 of 1   Show Feedback

Hide Code   Show CodeLens

```python
d = {'x': []}
print("starting the while loop")
while len(d.keys()) <= 2:
    print("number of keys in d:", len(d.keys()))
    print('Dictionaries')
    d['x'].append('d')
    print("number of keys in d after appending:", len(d.
print("end of the while loop")
```

Activity: 2 -- ActiveCode (ac14_11_3)

You have attempted 3 of 3 activities on this page