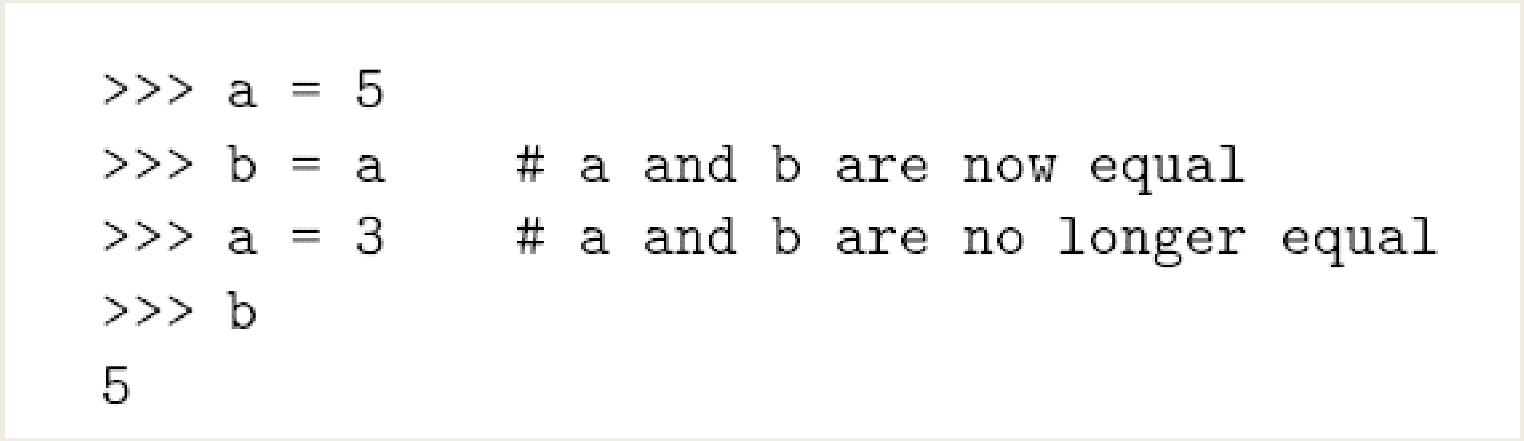
LECTURE 7: ITERATION

Hung-Yu Wei

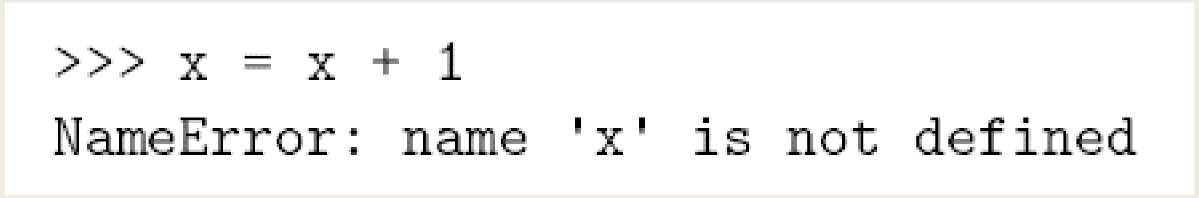
Assignment and Re-assignment

- =

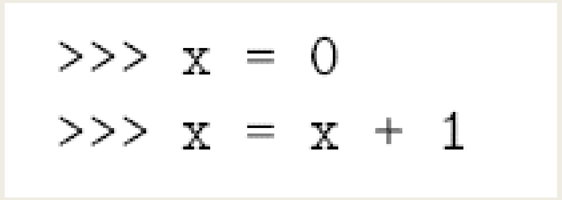


```
>>> a = 5
>>> b = a     # a and b are now equal
>>> a = 3     # a and b are no longer equal
>>> b
5
```

Updating variables



```
>>> x = x + 1
NameError: name 'x' is not defined
```



```
>>> x = 0
>>> x = x + 1
```

- Initialize the variable
- Update the variable

while

- While the condition is true, do the following things
  - *Continue these steps unless the condition is false*

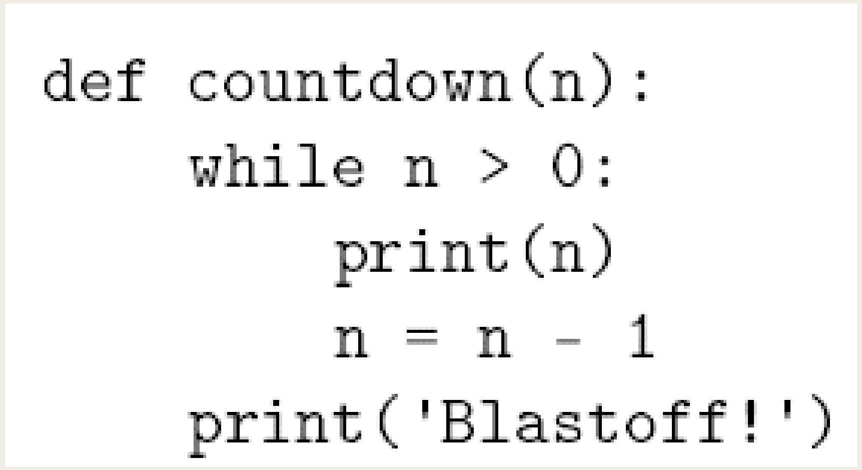
- **Syntax**

```
while condition :
    ... statements to execute ...
    ... more statements ...
```

- How the while loop is run
  1. *Determine whether the **condition** is true or false.*
  2. *If **false**, exit the while statement and continue execution at the next statement.*
  3. *If the condition is **true**, run the body and then go back to step 1.*

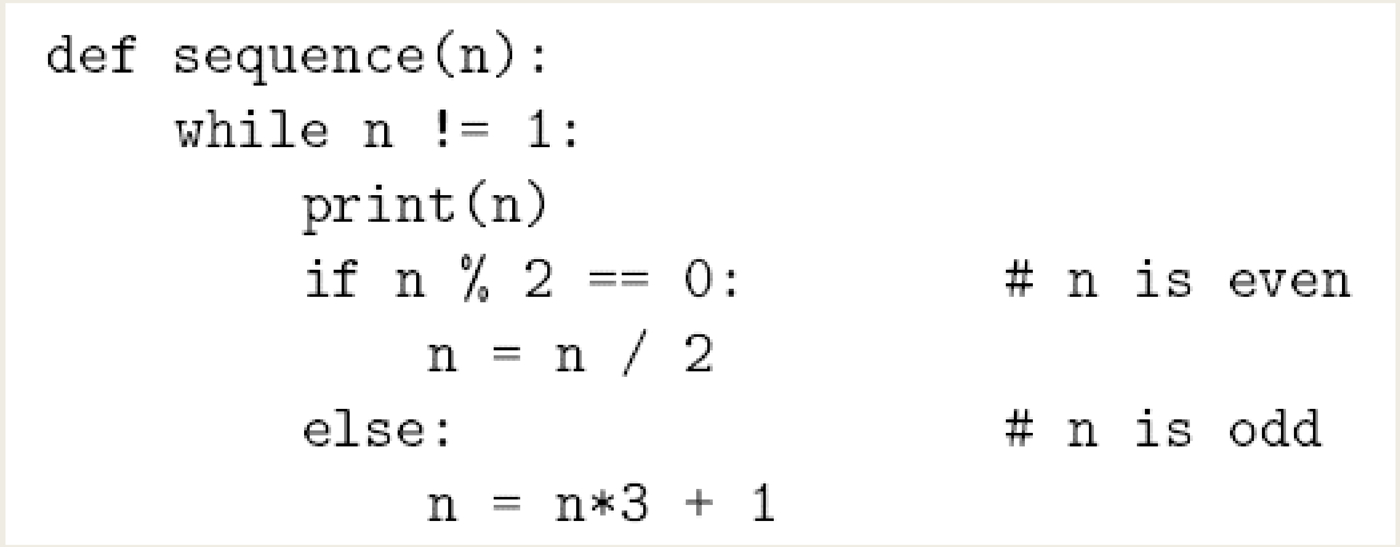
Example: while

- While n is greater than 0, display the value of n and then decrement n. When you get to 0, display the word Blastoff!



```
def countdown(n):
    while n > 0:
        print(n)
        n = n - 1
    print('Blastoff!')
```

Infinite loop



```
def sequence(n):
    while n != 1:
        print(n)
        if n % 2 == 0:           # n is even
            n = n / 2
        else:                    # n is odd
            n = n*3 + 1
```

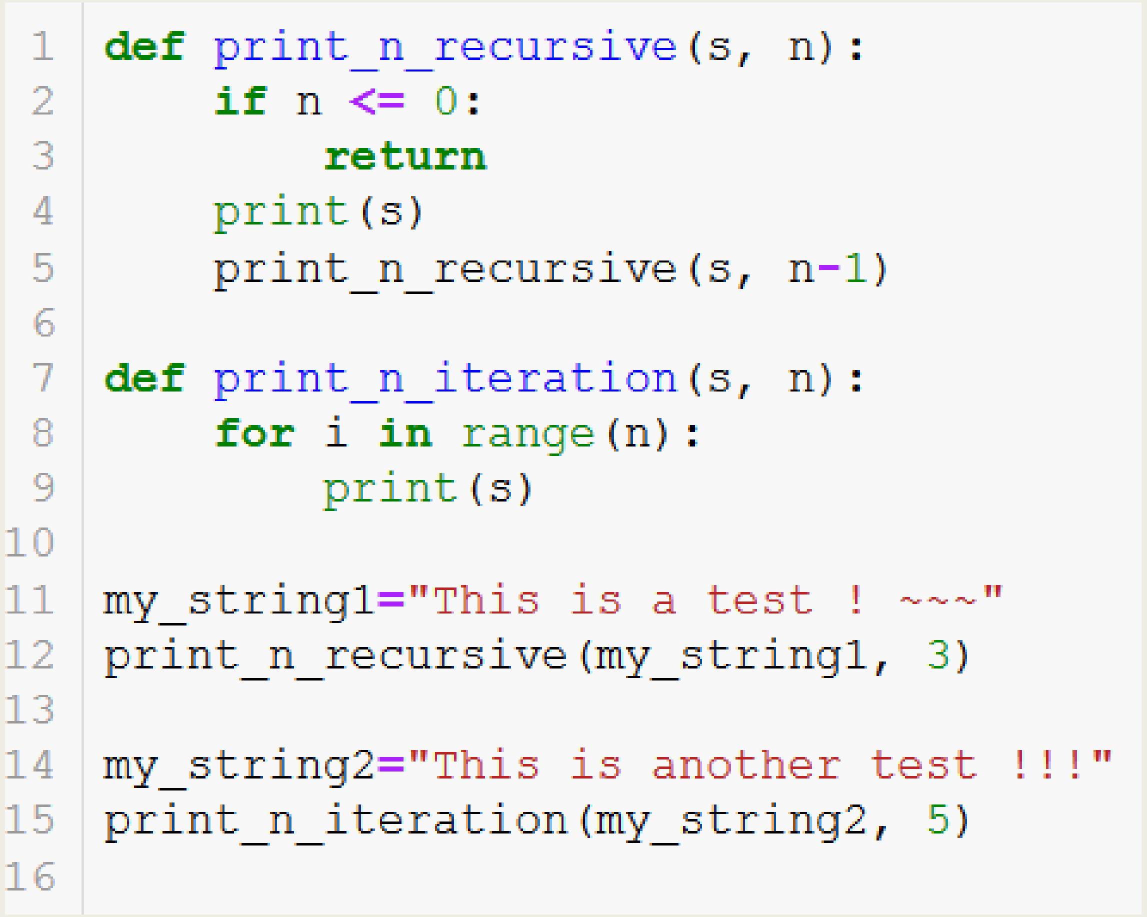
- A program never stops
  - *Be careful*
- Stop condition
  - *Will it occur eventually?*
- Update the variable to meet the stop condition

Nobody can prove or disprove that this will terminates for all cases

- Similar cases
  - *Iteration*
  - *Recursion*

Iteration v.s. Recursion

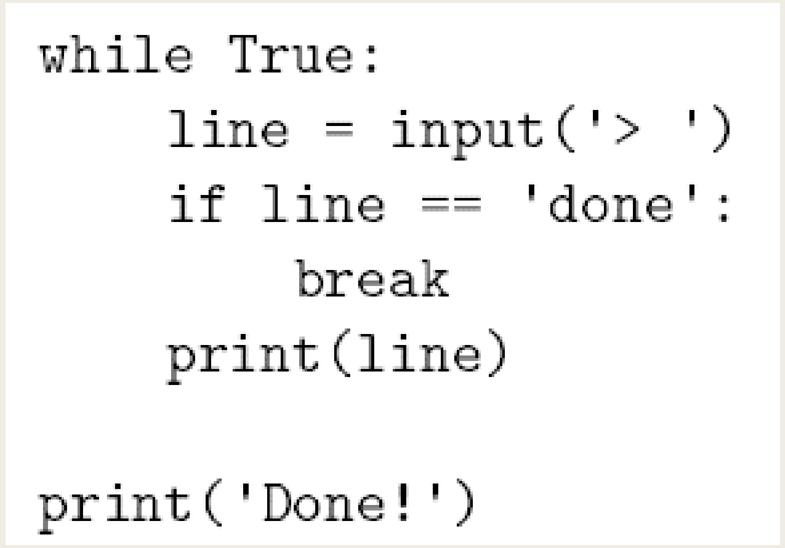
- Iteration (section 7.3)
- Recursion (section 5.8)
- Example
  - *Print a string for n times*



```python
def print_n_recursive(s, n):
    if n <= 0:
        return
    print(s)
    print_n_recursive(s, n-1)

def print_n_iteration(s, n):
    for i in range(n):
        print(s)

my_string1="This is a test ! ~~~"
print_n_recursive(my_string1, 3)

my_string2="This is another test !!!"
print_n_iteration(my_string2, 5)
```

break

- Jump out of a loop
  - *break*
  - *When condition meet, use break to jump out of loop*
- Express the stop condition affirmatively



```
while True:
    line = input('> ')
    if line == 'done':
        break
    print(line)

print('Done!')
```

Algorithm: Newton’s Method

- Algorithm: a mechanical process for solving a category of problems
- Newton’s Method
  - https://en.wikipedia.org/wiki/Newton's_method
- Finding square root of **a**

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$y = \frac{x + a/x}{2}$$

```
while True:
    print(x)
    y = (x + a/x) / 2
    if y == x:
        break
    x = y
```

```
if abs(y-x) < epsilon:
    break
```

Y and x should be close (but might not be equal)

Reading

- Chapter 7 in textbook “Think Python”