ENGG1811 Computing for Engineers

Week 3A: for, list processing, range, project

Lecture 3A

- The key topic today is the for-loop

- We will also do an in-class project which makes use of a few topics that you have learnt so far. These topics are:
  - List, for-loop, function, plotting

Why using loops in programming?

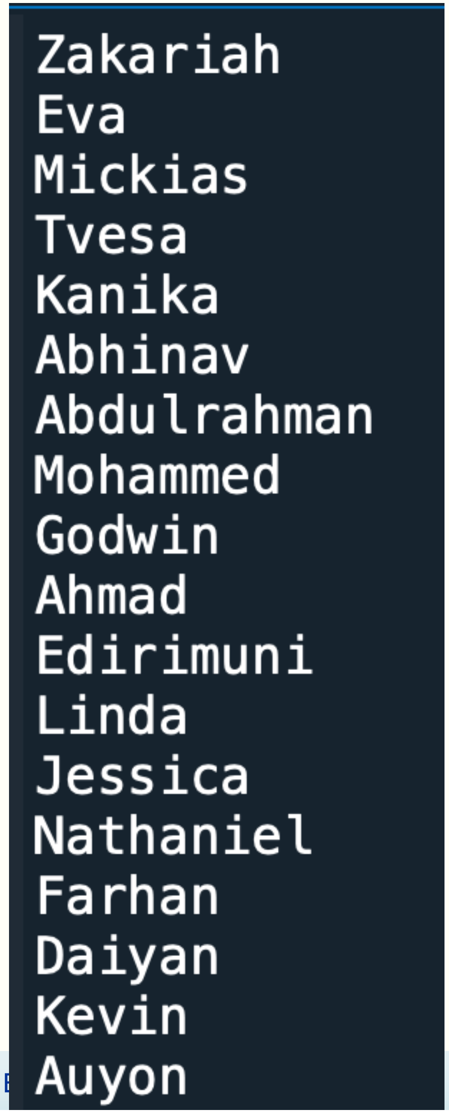
- Let us hear from Mark Zuckerberg (founder of Facebook) on why you need loops in programming
- https://www.youtube.com/watch?v=mgooqyWMTxk

Iteration (Repetition)

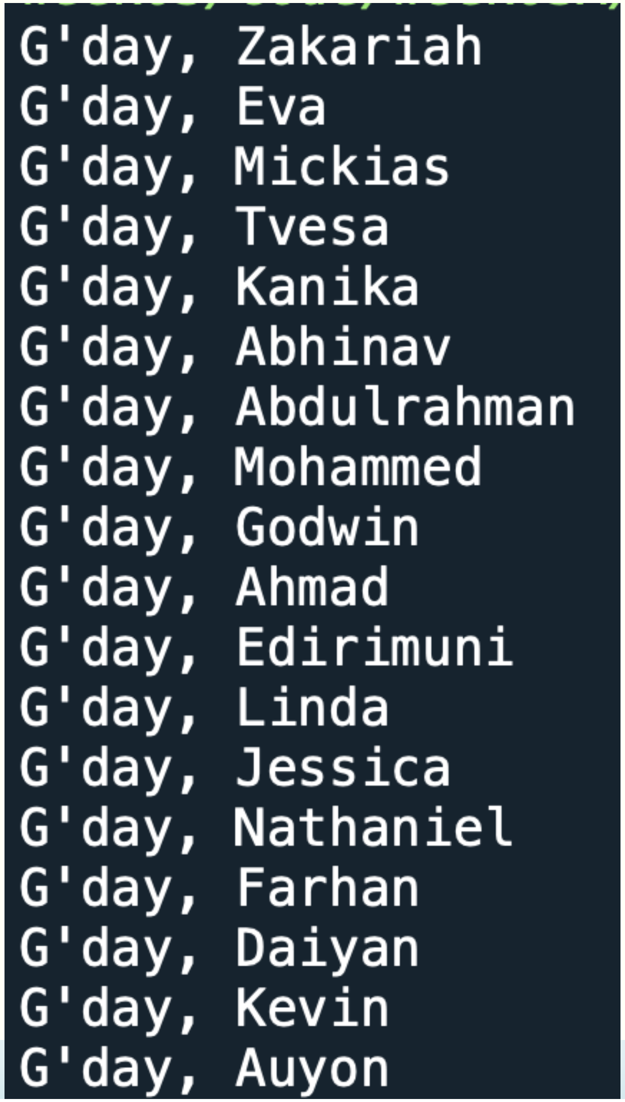
- Often need to execute statements repeatedly
- Loops are statements that can do this
- Process is called iteration
- Kinds of loop:
  - For (iterate a fixed number of times)
  - While (iterate as long as something is True)
- We will spend a part of the lecture in the next few weeks to learn about loops

G’day, mate!

- I wish to say G’day to the students in an ENGG1811 class
- I’ve created a list of names. There are 259 names.

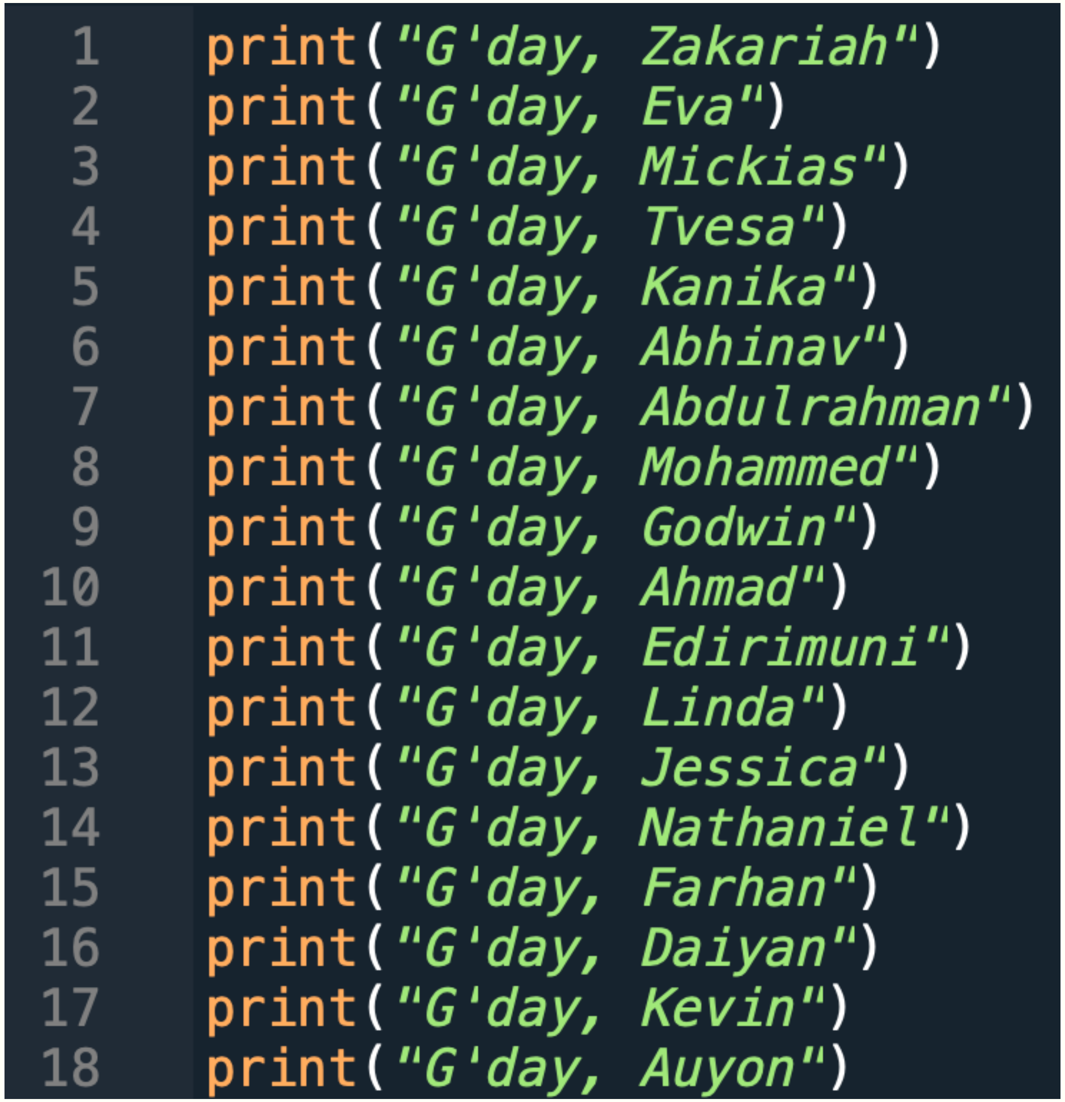


```
Zakariah
Eva
Mickias
Tvesa
Kanika
Abhinav
Abdulrahman
Mohammed
Godwin
Ahmad
Edirimuni
Linda
Jessica
Nathaniel
Farhan
Daiyan
Kevin
Auyon
```



```
G'day, Zakariah
G'day, Eva
G'day, Mickias
G'day, Tvesa
G'day, Kanika
G'day, Abhinav
G'day, Abdulrahman
G'day, Mohammed
G'day, Godwin
G'day, Ahmad
G'day, Edirimuni
G'day, Linda
G'day, Jessica
G'day, Nathaniel
G'day, Farhan
G'day, Daiyan
G'day, Kevin
G'day, Auyon
```

We can use the following code:

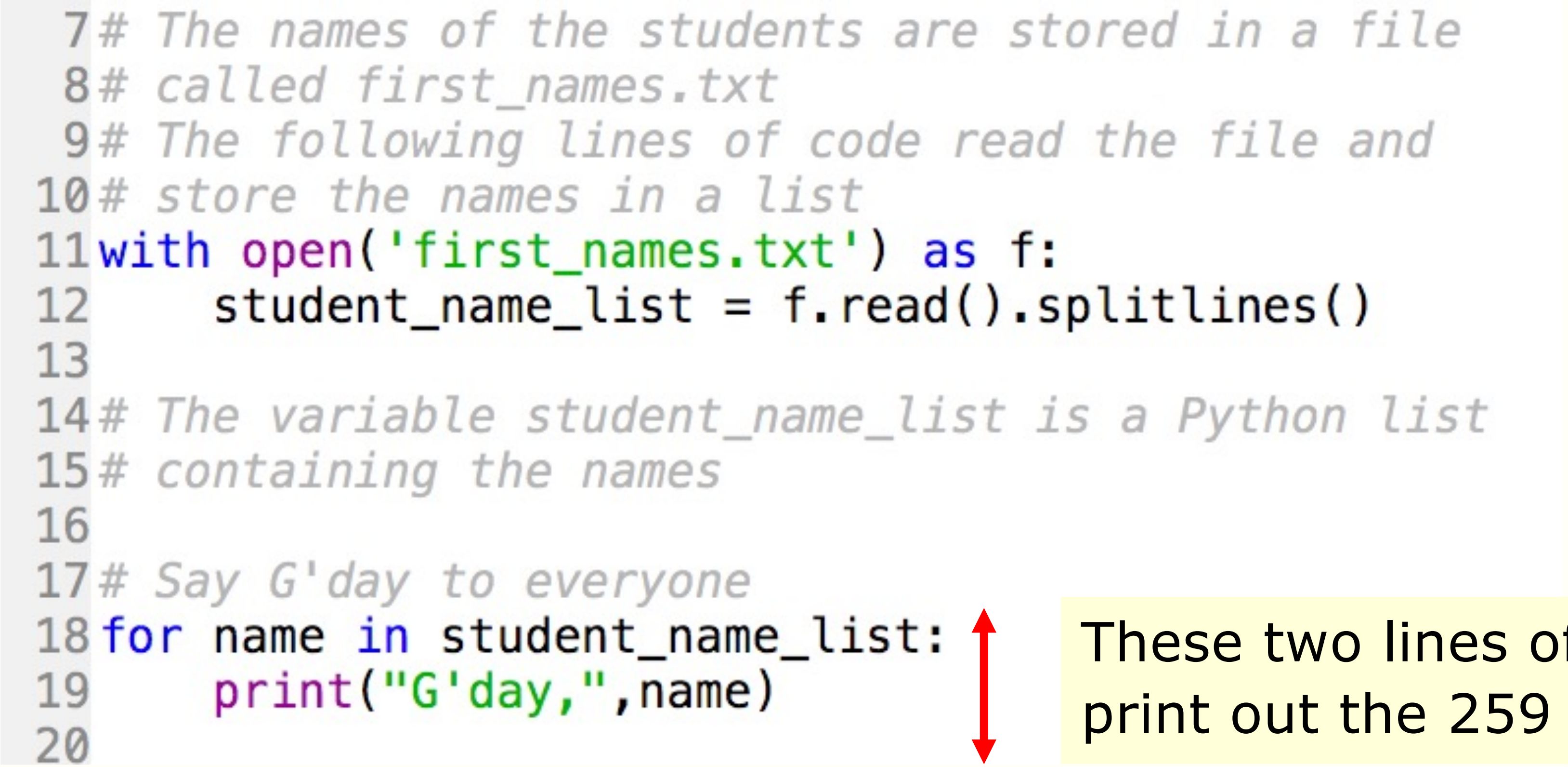


```python
print("G'day, Zakariah")
print("G'day, Eva")
print("G'day, Mickias")
print("G'day, Tvesa")
print("G'day, Kanika")
print("G'day, Abhinav")
print("G'day, Abdulrahman")
print("G'day, Mohammed")
print("G'day, Godwin")
print("G'day, Ahmad")
print("G'day, Edirimuni")
print("G'day, Linda")
print("G'day, Jessica")
print("G'day, Nathaniel")
print("G'day, Farhan")
print("G'day, Daiyan")
print("G'day, Kevin")
print("G'day, Auyon")
```

There are still 241 lines ☹

The enlightened way

- The code is in gday.py

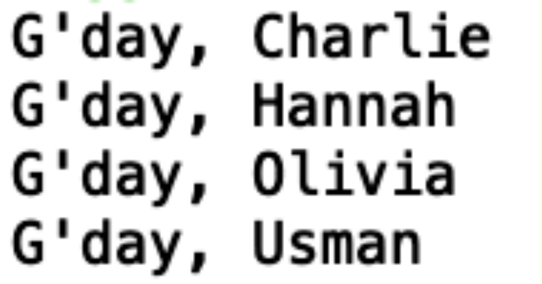


```
# The names of the students are stored in a file
# called first_names.txt
# The following lines of code read the file and
# store the names in a list
with open('first_names.txt') as f:
    student_name_list = f.read().splitlines()

# The variable student_name_list is a Python list
# containing the names

# Say G'day to everyone
for name in student_name_list:
    print("G'day,",name)

```

These two lines of code print out the 259 G’day

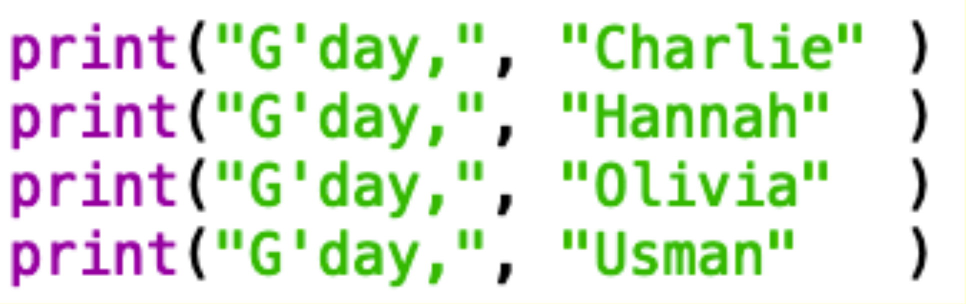
Writing for-loop

- End result wanted

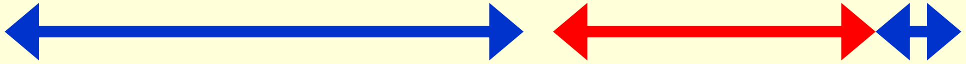


```
G'day, Charlie
G'day, Hannah
G'day, Olivia
G'day, Usman
```

- Long code



```
print("G'day,", "Charlie" )
print("G'day,", "Hannah"  )
print("G'day,", "Olivia"  )
print("G'day,", "Usman"   )
```

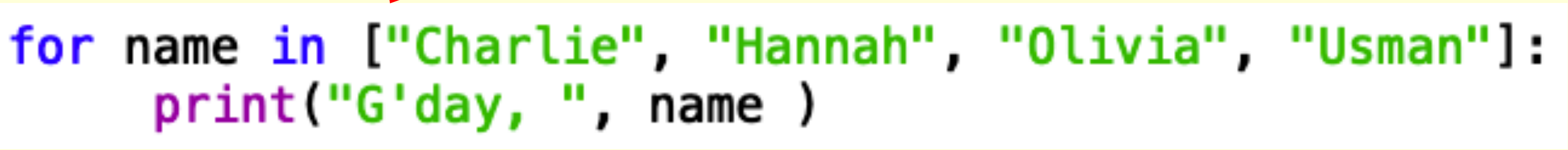


The same for each line

Vary for each line

- For loop

A list containing what is to be varied for each line



```
for name in ["Charlie", "Hannah", "Olivia", "Usman"]:
    print("G'day, ", name )
```

For loop

```
for name in ["Charlie", "Hannah", "Olivia", "Usman"]:
    print("G'day, ",name)
```

- The code is in gday_explained.py

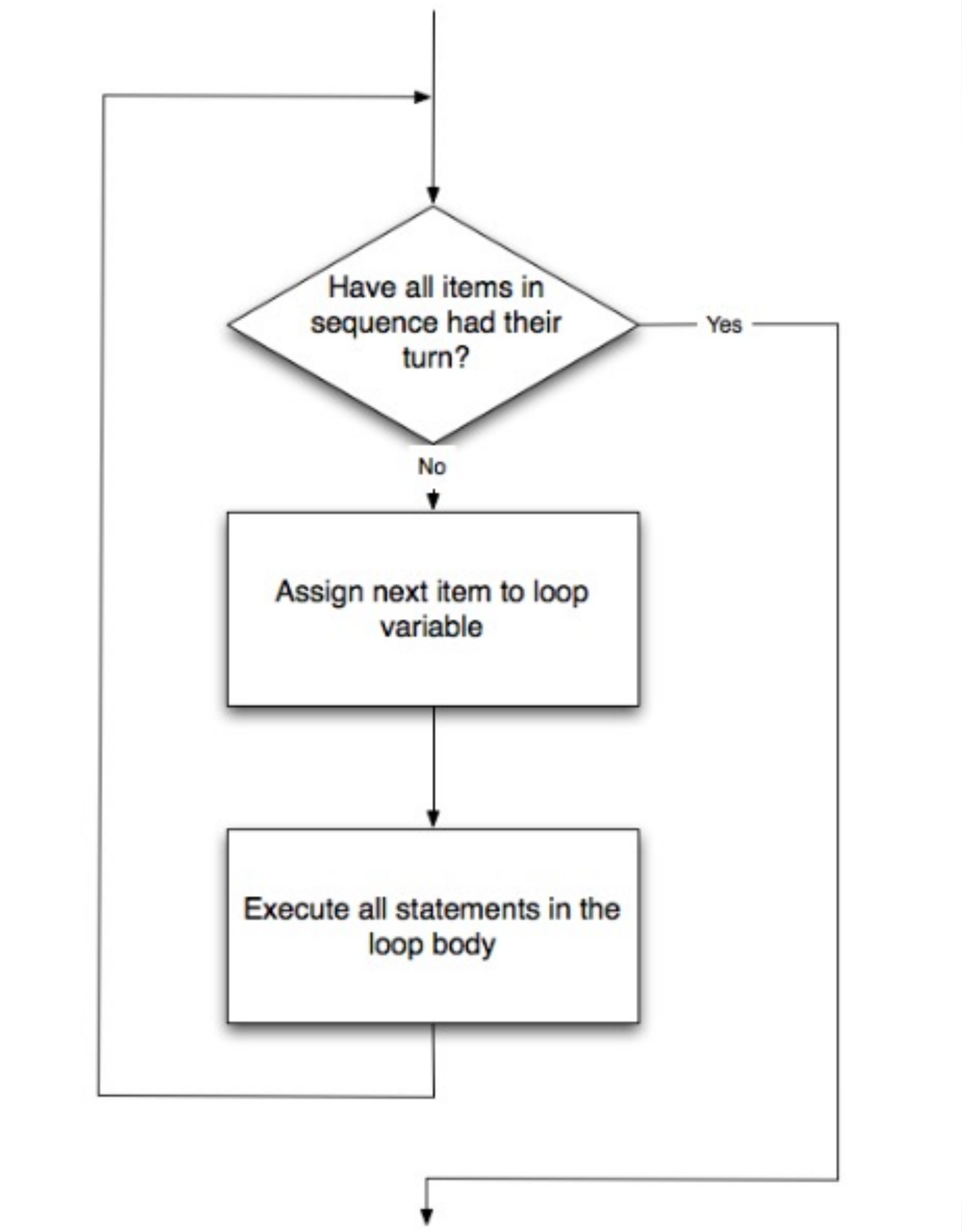
- Let us copy the code to Python Tutor and see how it is executed
- http://pythontutor.com/

The for-loop explained

```
for name in ["Charlie", "Hannah", "Olivia", "Usman"]:
    print("G'day, ",name)
```

- The variable **name** is called the loop variable
- Code under for-loop is indented
- The loop variable is assigned to the first item in the list
- **name** is now the string "Charlie”. The code in the for-loop is executed with the variable **name** having the value of “Charlie“
- After executing the code under the for-loop, the execution returns to the for-line. The computer checks whether there is a next item in the list. Yes, there is and the computer assigns "Hannah" to the variable **name.** The code in the for-loop is executed assuming this value of **name**
- This is repeated until all items in the list have been used

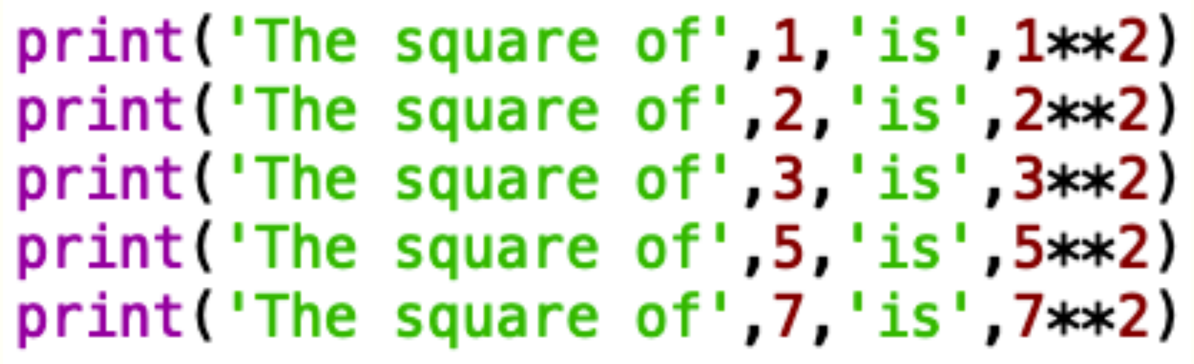
Flowchart


Have all items in sequence had their turn?
Yes
No
Assign next item to loop variable
Execute all statements in the loop body

http://interactivepython.org/runestone/static/thinkcspy/PythonTurtle/FlowofExecutionoftheforLoop.html

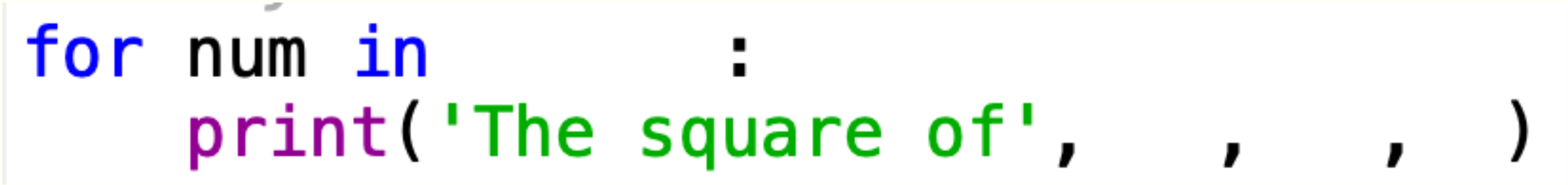
Exercise

- The file is for_exercise_prelim.py
- Use a for loop to replace the following five statements:



```
print('The square of',1,'is',1**2)
print('The square of',2,'is',2**2)
print('The square of',3,'is',3**2)
print('The square of',5,'is',5**2)
print('The square of',7,'is',7**2)
```

- To get started:



```
for num in          :
    print('The square of',   ,   ,   )
```

Using for-loops to create a list from another list

- Very often you may need to create a list from another list
- For example, you are given the list

  [2, -3, 4, -5]

  and you want to compute the cube of each

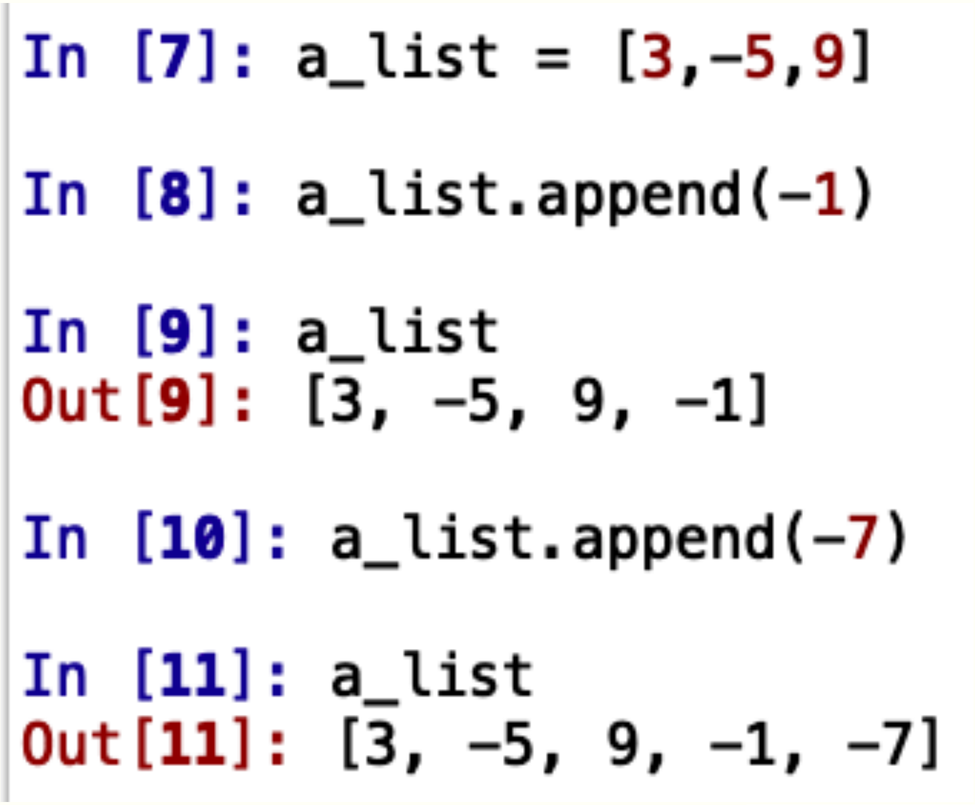
  number and store the results in a new list, which is:

  [8, -27, 64, -125]

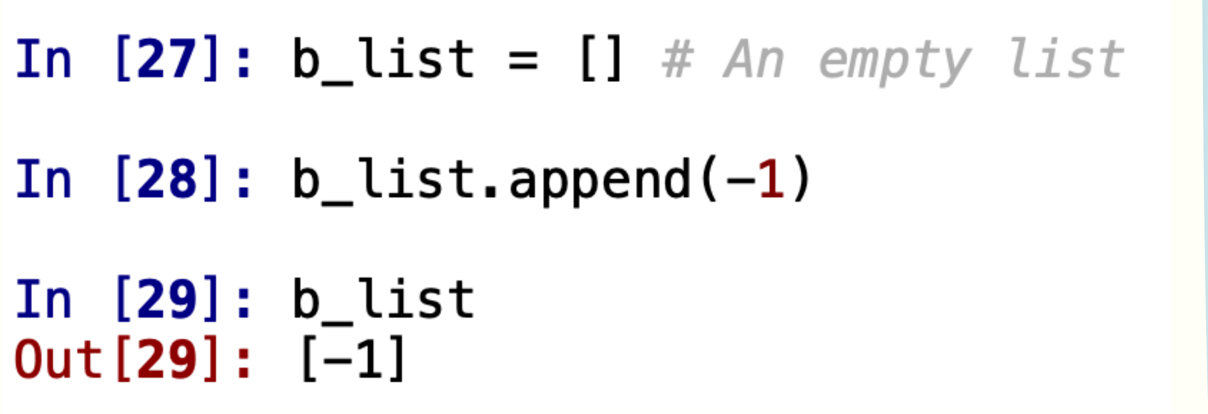
- There are two methods you can do this. We will use .append() today.

- Let us first understand what .append() does first

Appending an element to a list



```
In [7]: a_list = [3,-5,9]

In [8]: a_list.append(-1)

In [9]: a_list
Out[9]: [3, -5, 9, -1]

In [10]: a_list.append(-7)

In [11]: a_list
Out[11]: [3, -5, 9, -1, -7]
```



```
In [27]: b_list = [] # An empty list

In [28]: b_list.append(-1)

In [29]: b_list
Out[29]: [-1]
```

Example: Create a list from another list (1)

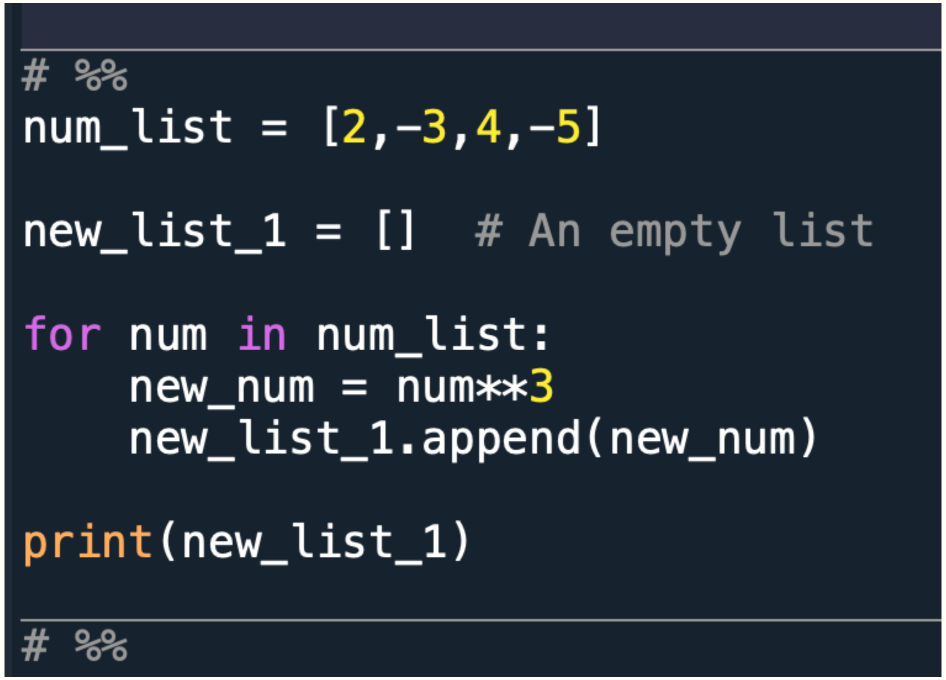
- Use the list [2, -3, 4, -5] to create the new list [8, -27, 64, -125] using .append()

```
num_list = [2, -3, 4, -5]

new_list_1 = []  # An empty list

for num in num_list:
    new_num = num**3
    new_list_1.append(new_num)
```

- Code in the first cell in create_list_from_list_prelim.py
- Visualize with Python tutor http://pythontutor.com/

Cells in Spyder

- Spyder allows us to divide the code into cells
  - Cells are separated by a line of # %%



```
# %%
num_list = [2,-3,4,-5]

new_list_1 = []  # An empty list

for num in num_list:
    new_num = num**3
    new_list_1.append(new_num)

print(new_list_1)

# %%
```

A cell

- and we can run the code in each cell independently
  - Good for testing and debugging code
  - To run a cell, make sure your mouse cursor is in that cell and click "Run Current Cell"

Example: Create a list from another list (2)

```
for num in num_list:
    new_num = num**3
    new_list_1.append(new_num)
```

```
for num in num_list:
    if num > 0:
        new_num = num**3
    else:
        new_num = num**2
    new_list_2.append(new_num)
```

- The operation performed on each element of the list.
- We can make it more complicated.
  - Example: If num > 0, compute its cube; otherwise, square it

- Code in the second cell in create_list_from_list_prelim.py

Example: Create a list from another list (3)

```
for num in num_list:
    if num > 0:
        new_num = num**3
    else:
        new_num = num**2
    new_list_3.append(new_num)
```

- We can move these lines of code into a function and call the function within the loop

- Code in the third cell in create_list_from_list_prelim.py
  - Will complete it in class

Operations on list

- You know how to append an element to a list
- There are other operations that you can do on a list
  - Finding the maximum or minimum element in a list
  - Sum the elements in a list
  - Determining the number of elements in a list
    - Terminology: length of a list = number of elements in a list
  - See list_processing.py
  - There are many other operations:
    - E.g. sort, count the occurrence of a value etc.
    - See https://www.programiz.com/python-programming/methods/list

range()

- You may want to create a list of square numbers [0,1,4,9,16,25,36]. You can use

```
sq_list = []
for k in [0,1,2,3,4,5,6]:
        sq_list.append(k**2)
```

Is there a faster way than to write [0,1,2,3,4,5,6]?

```
sq_list = []
for k in range(7):
        sq_list.append(k**2)
```

range(7) produces 7 integers starting from 0

Code: range_ex.py

range()

- range() is a Python function that generates a sequence of integers
- The function can take 1 to 3 inputs and its behaviour depends on the number of inputs
- Examples in range_ex.py

range() expression	sequence	explanation
range(5)	0,1,2,3,4	One input. Starting from 0. Keep increasing by 1. Does not including the number specified by the input.
range(2,8)	2,3,4,5,6,7	Two inputs. 1st number in list = 1st input

- With 2 inputs, the function has the form range(start,stop)
  - range(0,stop) is the same as range(stop)
- #elements in the list = stop - start

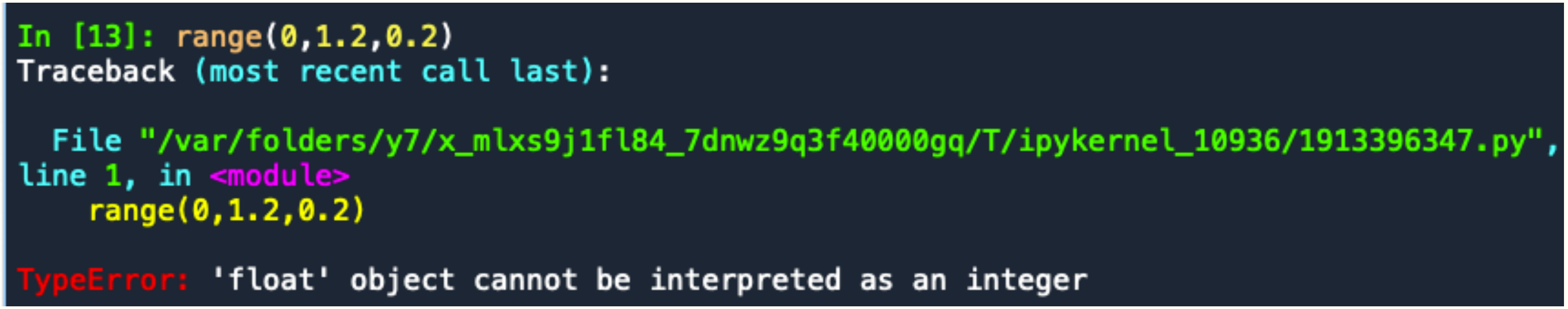
range()

range() expression	sequence	explanation
range(2,20,4)	2,6,10,14,18	The first input (=2 in this example) is the starting value of the sequence. The last input (= 4 in this example) is the increment. The next element of the sequence is obtained by adding the increment to the element before: 2, 2 + 4, 2 + 4 + 4 Keep incrementing until a number >= the last input (= 20 in this case) is reached. Stop but don't include the last number generated.

- The general form is range(start,stop,inc)
- #elements in the list = ceil ((stop-start)/inc)
  - ceil(x) = smallest integer greater than or equal to x

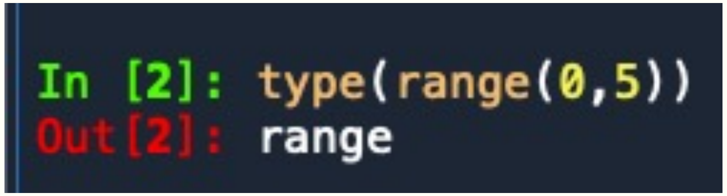
Some notes on range()

- The function range() only accepts integers as its inputs
  - If any one of the inputs to range() is not an integer, you get an error.
  - There is function in a Python package which is similar to range() but allows non-integral inputs. You will learn that in a few weeks' time.



```
In [13]: range(0,1.2,0.2)
Traceback (most recent call last):

  File "/var/folders/y7/x_mlxs9j1fl84_7dnwz9q3f40000gq/T/ipykernel_10936/1913396347.py",
line 1, in <module>
    range(0,1.2,0.2)

TypeError: 'float' object cannot be interpreted as an integer
```

- The output of range() is not a list



```
In [2]: type(range(0,5))
Out[2]: range
```

Some notes on range() (continued)

- Typically, we use range() together with a for-loop to produce another list, e.g.

```
sq_list = []
for k in range(7):
        sq_list.append(k**2)
```

Project: goal

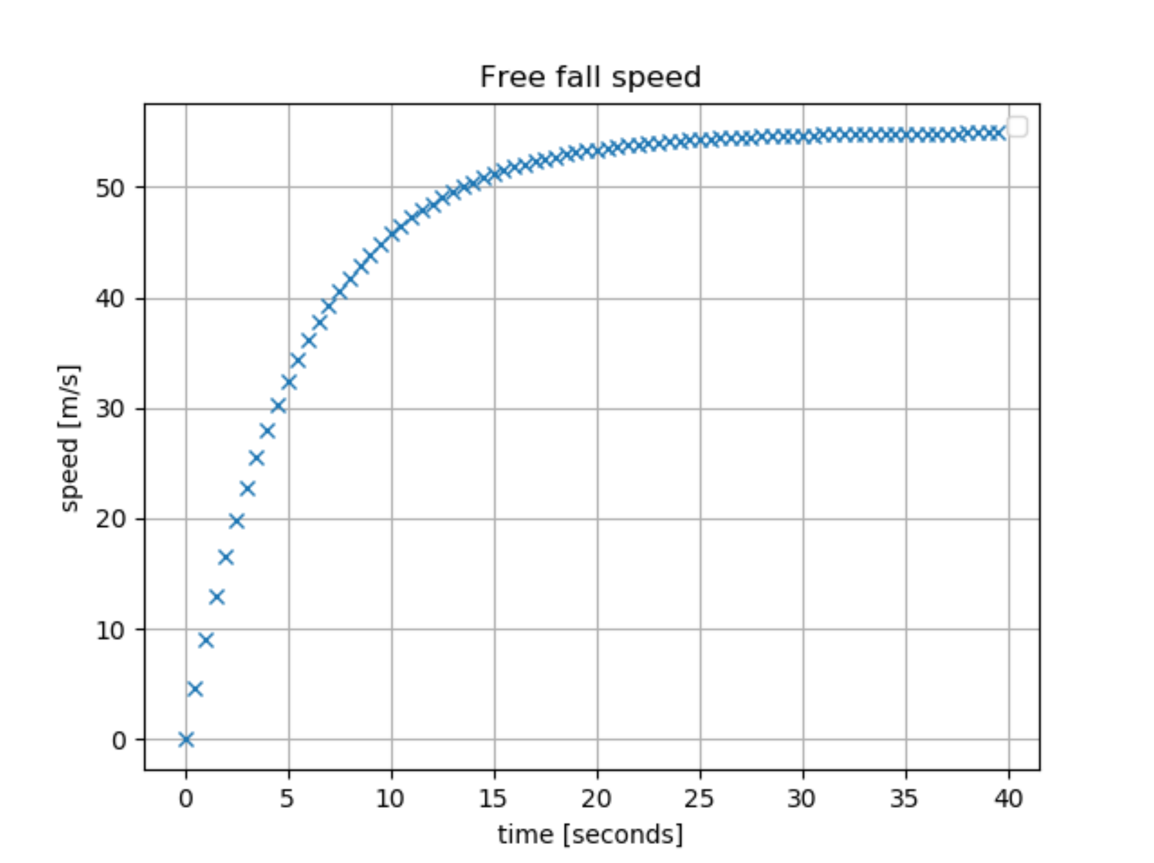
- If you drop an object of mass *m* in a medium with drag coefficient *d* and acceleration due to gravity *g*, then the object's speed *v(t)* at time *t* is given by:

$$v(t) = \frac{gm}{d}\left(1 - e^{-\frac{d}{m}t}\right)$$

- Given the numerical value of *m*, *g* and *d*, the goal of the project is to plot *v(t)* against *t*
  - for t = 0, 0.5, 1, 1.5, …., 39.5, 40
- You certainly know how to do this by using pen, paper and calculator. You may also need a bit of perseverance because it does get a bit repetitive

Project: end product

- You will do it in Python
- The end product



Part 1: Write a function

- mass *m*, drag coefficient *d*, acceleration due to gravity *g*
- speed *v(t)* at time *t* is:

$$v(t) = \frac{gm}{d}\left(1 - e^{-\frac{d}{m}t}\right)$$

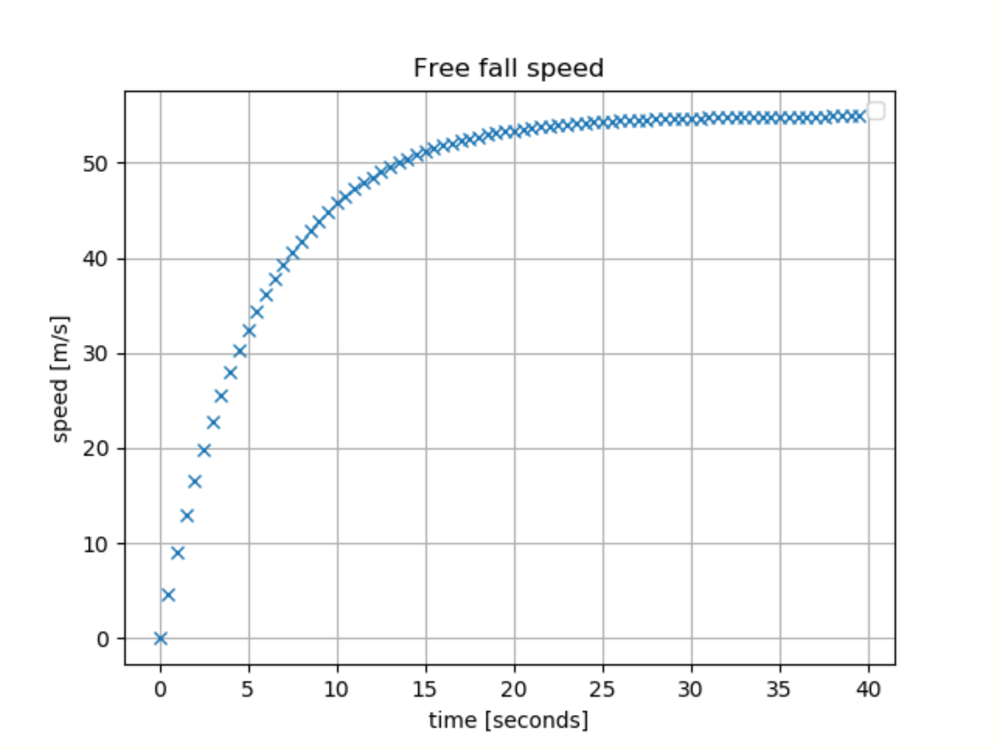
- We want a function which computes the speed *v(t)* for a given *t*
  - Open the file project_prelim.py
  - The function called free_fall() computes *v(t)*
    - The def line of the function is given in Line 16:

```
def free_fall(t, mass, drag):
```

    - We have done this for you already!

Part 2: Producing the graph

- You want to plot a graph of the free fall speed against time
- In order to produce the graph, you need to create two lists



List of time instants

- The first list is a list of time instants (in seconds). We ask you to use:

```
[0  0.5  1  1.5  2  2.5                                39.5  40]
```

- There are 81 numbers in the list and of course you are not going to type these 81 numbers in
- The function range() cannot be used directly because range() can only generate a sequence of integers, it cannot generate numbers with decimal points
  - range(0,40,0.5) will give an error

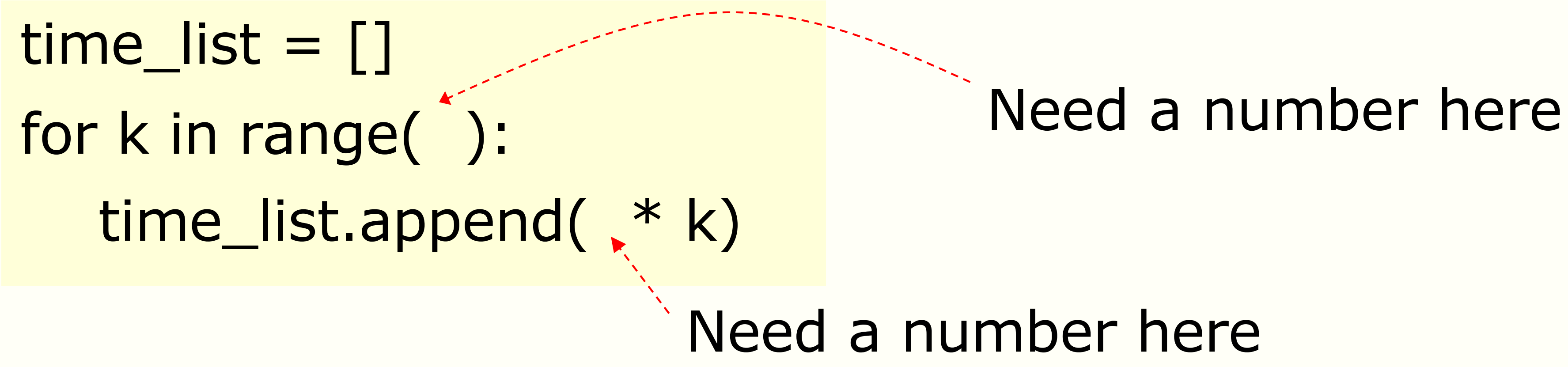
- Hint on the next page

Hint

- You can use range() to help you to produce this list:

```
[0  0.5  1  1.5  2  2.5                                   39.5  40]
```

- The hint is:



```
time_list = []
for k in range(   ):
    time_list.append(  * k)
```

Need a number here

Need a number here

range() gives: 0 1 2 3

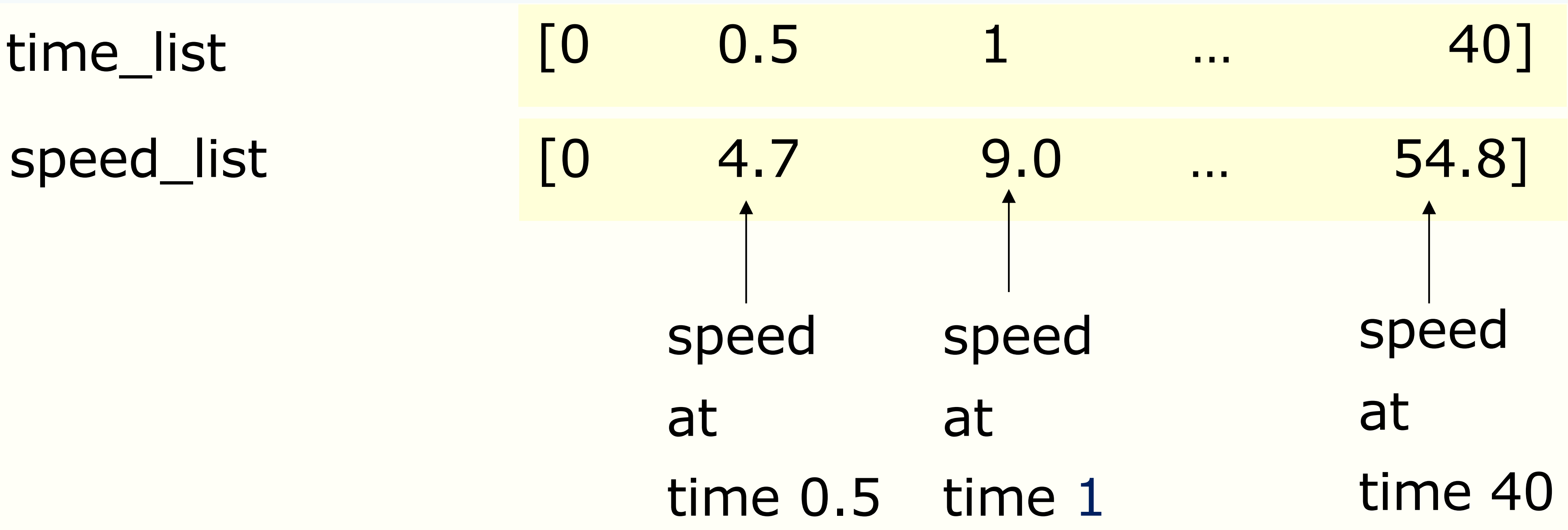
You want: [0 0.5 1 1.5 39.5 40]

List of speeds

- The second list is a list of speeds
- If you do this *manually*, you will do:
  - Time is 0. Use the speed formula. Speed = 0.
  - Time is 0.5. Use the speed formula. Speed = 4.692400935
  - Time is 1. Use the speed formula. Speed = 8.98399681455

  - Time is 40. Use the speed formula. Speed = 54.8885179036
- Of course, you aren't going to do the manual way since you have seen the trick

End results: two lists



time_list	[0	0.5	1	…	40]
speed_list	[0	4.7	9.0	…	54.8]
		speed at time 0.5	speed at time 1		speed at time 40

- You should use the list of times and the function free_fall()
- File project prelim.py
  - Lines 26-27: Complete the code for calculating time_list
  - Lines 32-33: Add the code for calculating speed list

Summary

- for-loop
  - To repeatedly do some actions
- List processing
- range()

- The time-speed trajectory project as an example of using programming to automate a mundane task