

# From zero to one: an introduction to programming with Python

Zhou (Yuen) Fang

Institute for Artificial Intelligence, University of Bremen

[yzfang1@gmail.com](mailto:yzfang1@gmail.com)

“Ask a question and you might look like a fool for 5 minutes.  
Don’t ask and you might be a fool forever”



# Why programming?



# Why python?

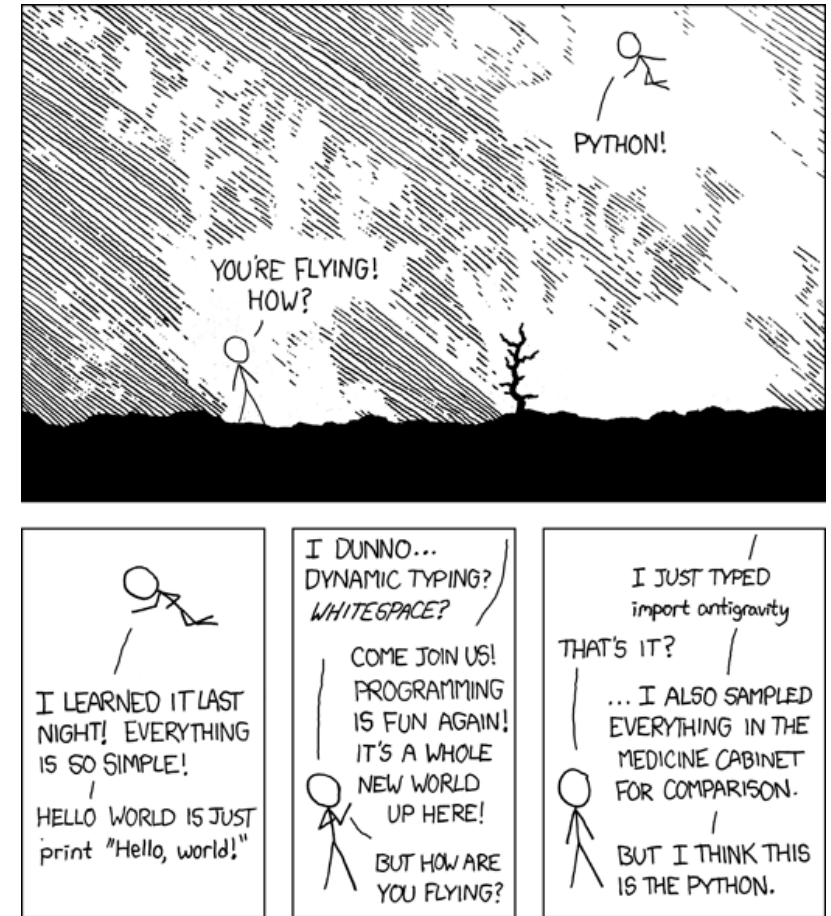
**JAVA:**

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, world!");  
    }  
}
```

**PYTHON:**

```
print 'Hello, World!'
```

[https://en.wikipedia.org/wiki/List\\_of\\_Python\\_software](https://en.wikipedia.org/wiki/List_of_Python_software)



# Overview

- Your Python environment
- Core concepts
- Reading data
- Visualizing data
- Basics of machine learning
- Predicting outcomes
- The art of debugging
- Questions?

# Your Python

- Python 2 vs 3
- Download and install Python distribution:  
<https://www.continuum.io/downloads>
- Download and install IDE (optional):  
<https://www.jetbrains.com/pycharm/download/>
- Tell IDE where the python interpreter is: File > Settings > Project > Python Interpreter > [your python path]

# Core concepts

- Data types, variables
- Control flow, loops
- Functions
- Comments

# Things in the Python universe (a.k.a. data types)

- Booleans True/ False
- Numbers (int, float, long, complex) 1, 1.0, 1L, 1+1j
- String "Mmm, food"
- List [12, 'watermelon', 'pieces']
- Tuple ('pears', 'and', 'apples')
- Dictionary {'easy': 'peasy', 'lemon': 'squeezy'}
- None None



# Things in the Python universe (2)

## Exercises

- What are the results of these operations? (answer and type)
  - $2+2$
  - $2+3.5$
  - $2>3$
  - "apple" + "pie"
  - $3/2.0$
  - $3/2$
  - "Give me " +  $2$  + " bananas"
  - "Give me " + `str(2)` + " bananas"

Not sure? Try it in your Python shell!

Use `type()` to see the type, for example: `type(3/2)`

# Naming things in the universe (a.k.a. variables)

- In real life: “Yuen” points to the person talking to you right now
- In Python: use the “=” sign to name entities
  - `a = 2`
  - `b = "pie"`

Try the previous commands:

```
a+a, a+3.5, a>3, "apple"+b, "give me "+a+" bananas"
```

- Assign the answer to `a+a` to a new variable `c`.
- What was the answer? Print it.
- What’s the type of the answer?

# Naming things in the universe (2)

## Variable (re)assignment

- Variables are variable: you can reassign a variable (name) to another thing in the universe
  - $a = 5$
  - $a = 6$
  - $a = a + 1$
- A variable has to be assigned to a value before it can be used
- A name cannot refer to two different things, that would be confusing! (But... see scope later)

# Control flow

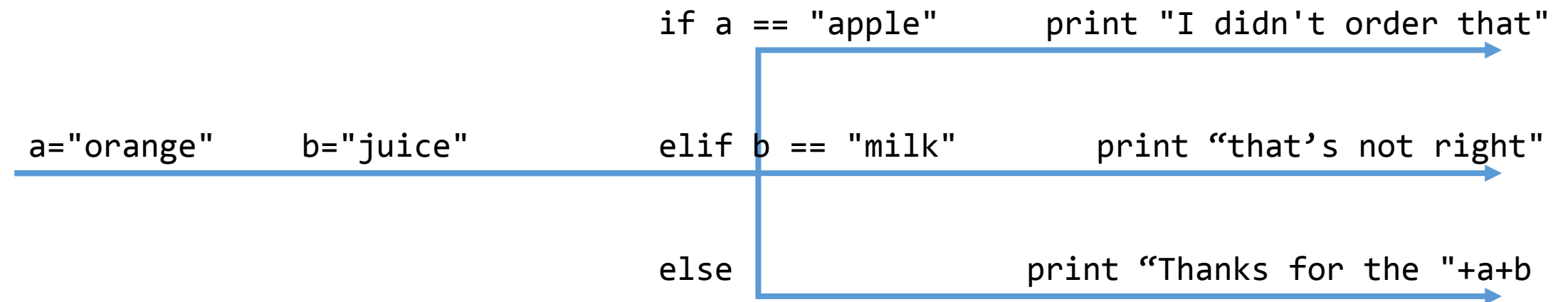
Until now:

`a="orange"      b="juice"      yummy=a+b      print yummy      ...`



- Conditional:

`a="orange"      b="juice"`



```
if a == "apple"      print "I didn't order that"
elif b == "milk"      print "that's not right"
else      print "Thanks for the "+a+b
```

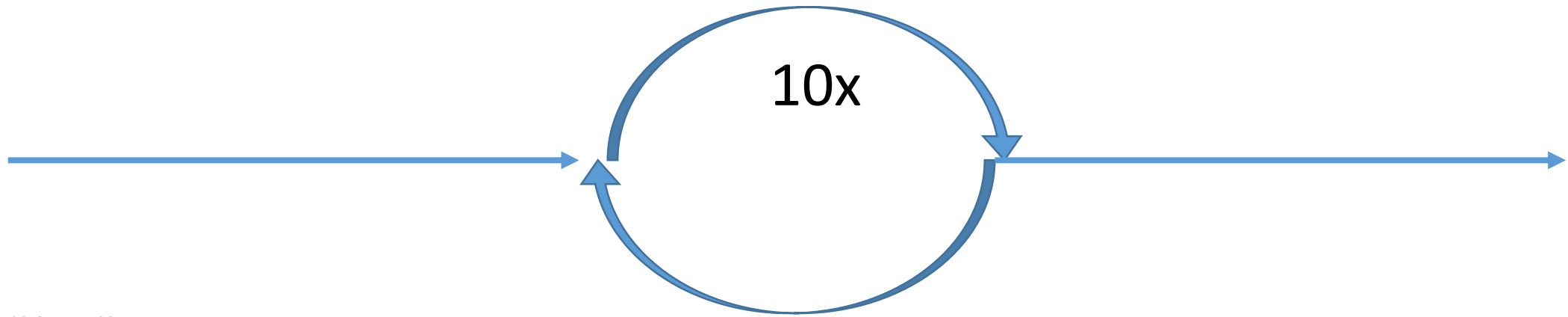
# Control flow (2)

What does this do if *oranges*=6 and *fries*=5?

```
if oranges == fries:
    print "balanced diet!"
elif oranges > fries:
    print "I like oranges!"
else:
    print "we need more fruit!!"
print "I'm done!"
```

Note: be careful of the ":" and indentation!

# Loops



“for” loop:

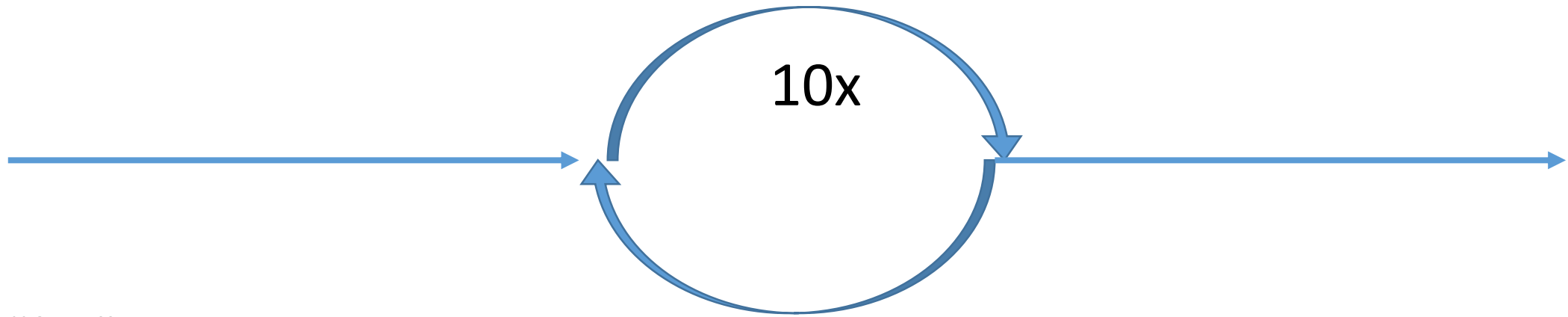
```
fruitsalad = ['apple', 'grape', 'melon',  
             'strawberry']
```

```
for item in fruitsalad:  
    print item+"juice"  
print "I made juice!"
```



```
applejuice  
grapejuice  
melonjuice  
strawberryjuice  
I made juice!
```

# Loops (2)



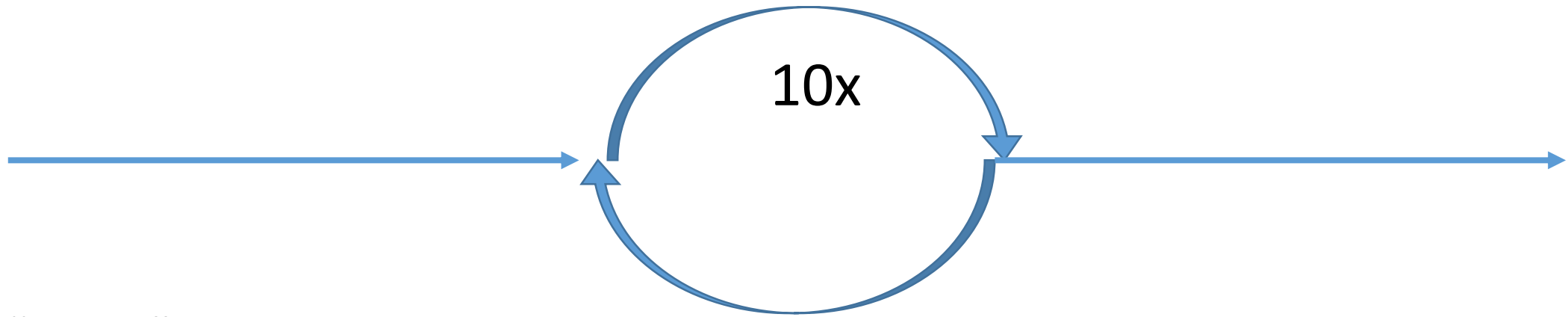
“for” loop:

```
a = "oranges"
for counter in range(10):
    print str(counter)+" "+a
print "So many oranges!"
```



```
0 oranges
1 oranges
[...]
8 oranges
9 oranges
So many oranges!
```

# Loops (3)



“while” loop:

```
counter = 0
while counter < 5:
    print "Counting to " + counter
    counter = counter + 2
print "Enough counting already!"
```



```
Counting to 0
Counting to 2
Counting to 4
Enough counting already!
```



# Functions

- Reusing code by giving them a name
- Defining a function:

```
def eat_watermelon(watermelon_size):  
    """eat 20% of the watermelon I've been given and return what's  
    left over """  
    remaining = watermelon_size * 0.8  
    return remaining
```

```
result = eat_watermelon(1.0)
```

- Existing functions we've used: `type()`, `str()`, `range()`

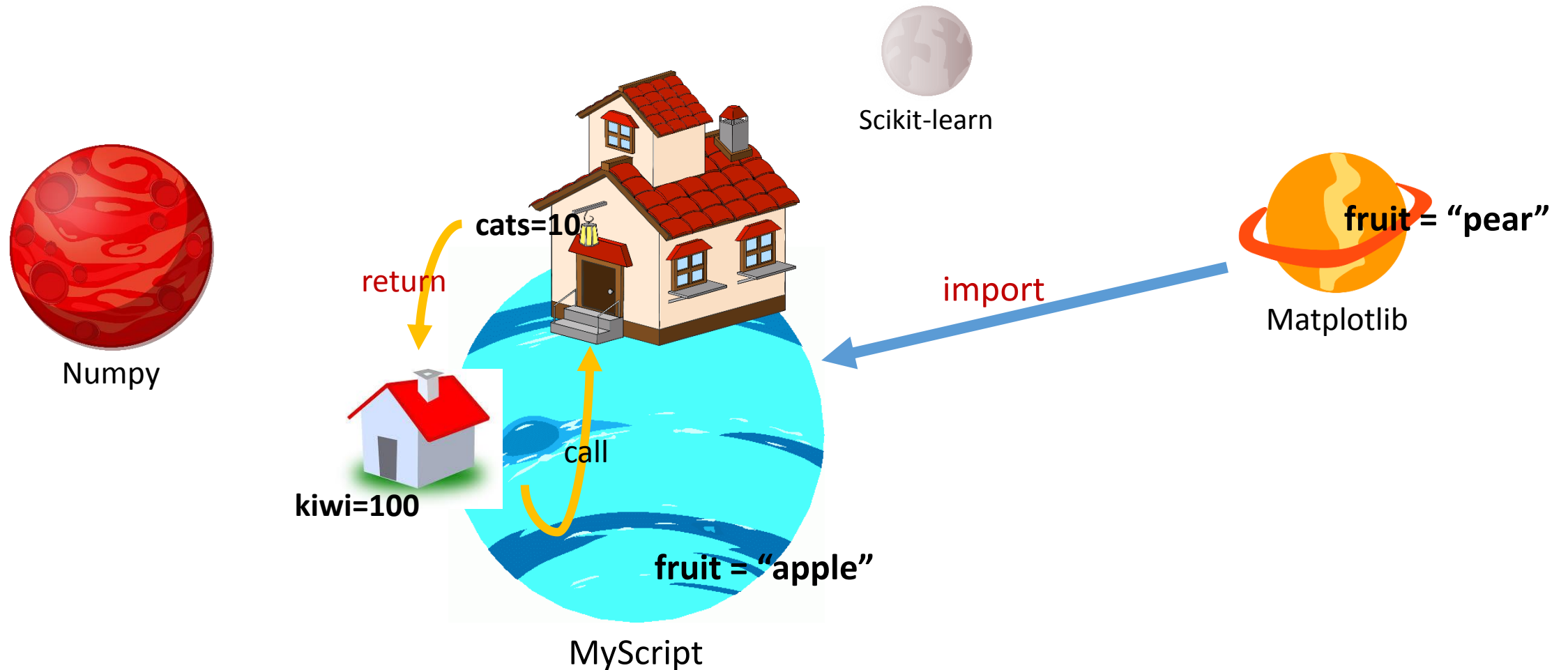
# Comments

- Describe what a function does by using triple quotation marks for opening and closing the comment
- Within your code, add comments using “#”. Describe what each chunk of code is supposed to do unless it’s very obvious.
- Comments help organize your thoughts and helps you debug/ come back to the code later! Keep in mind that what might seem obvious to you now, might not be obvious later.

# Using modules/ libraries

- People provide software packages for others to use
- Preferably use a package manager to install packages
- We'll use matplotlib, numpy and scikit-learn packages. These are pre-installed in the Anaconda distribution.
- Use “**import**” keyword

# Worlds in the Python universe (a.k.a. scope)



# Reading data

- Download data from <https://github.com/zyfang/AdaLovelace2016>
- Use numpy module to read data and represent it as a kind of “list”

```
import numpy  
  
my_data = numpy.genfromtxt('C:/Users/Yuen/Desktop/ada_data.csv',  
dtype=None, names = True, delimiter=',')
```

- Have a look at the data by inspecting it:

```
print my_data.size  
print my_data.dtype.names  
print my_data[0]  
print my_data["NCD_0"]
```

# Visualizing data

- Whenever possible, visualize data to get a feel for what the data actually looks like

```
import matplotlib.pyplot as plt
```

```
plt.scatter([1,2,3,4], [2,4,6,8])
```

```
plt.show()
```

```
plt.scatter(my_data["NCD_0"], my_data["AI_0"])
```

```
plt.xlabel("Number of Created Discussions t0")
```

```
plt.ylabel("Author Increase t0")
```

```
plt.show()
```

# Visualizing data (summary of abbreviations in dataset)

Number of Created Discussions (NCD)

Author Increase (AI)

Attention Level (measured with number of authors) (AS(NA))

Burstiness Level (BL)

Number of Atomic Containers (NAC)

Attention Level (measured with number of contributions) (AS(NAC))

Contribution Sparseness (CS)

Author Interaction (AT)

Number of Authors (NA)

Average Discussions Length (ADL)

Number of Active Discussions(NAD)

Popularity (Annotation)

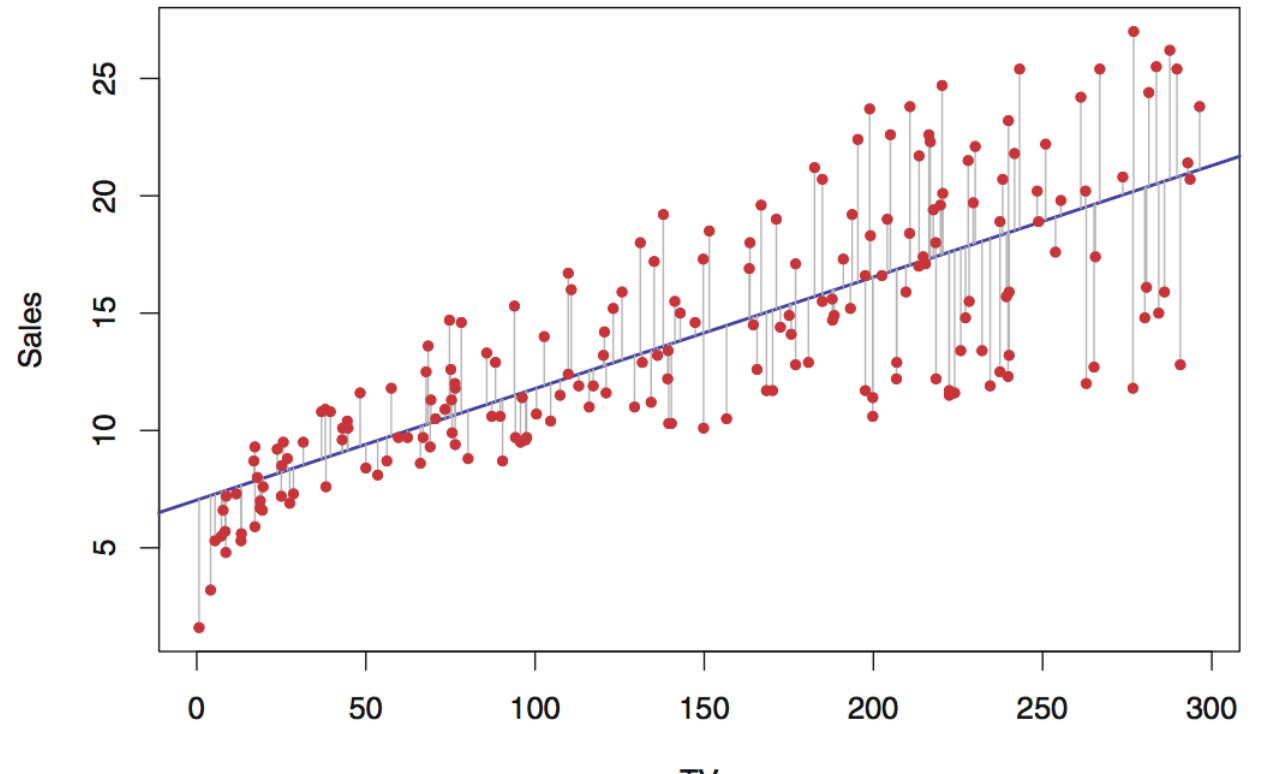
# Basics of machine learning

- “gives computers the ability to learn without being explicitly programmed”
- Typical learning problem: given a set of  $n$  [samples](#) of data, describe or predict properties of unknown data
- Supervised vs. Unsupervised
- Classification vs. Regression
- Training set and test set



# Linear Regression

- Simple linear regression: try to describe the data as a linear relation between  $x$  and  $y$
- $\text{Error} = \sum (y_{\text{actual}} - y_{\text{predicted}})$
- $y_{\text{predicted}} = \alpha + \beta x$
- Minimize error



# Linear Regression (2)

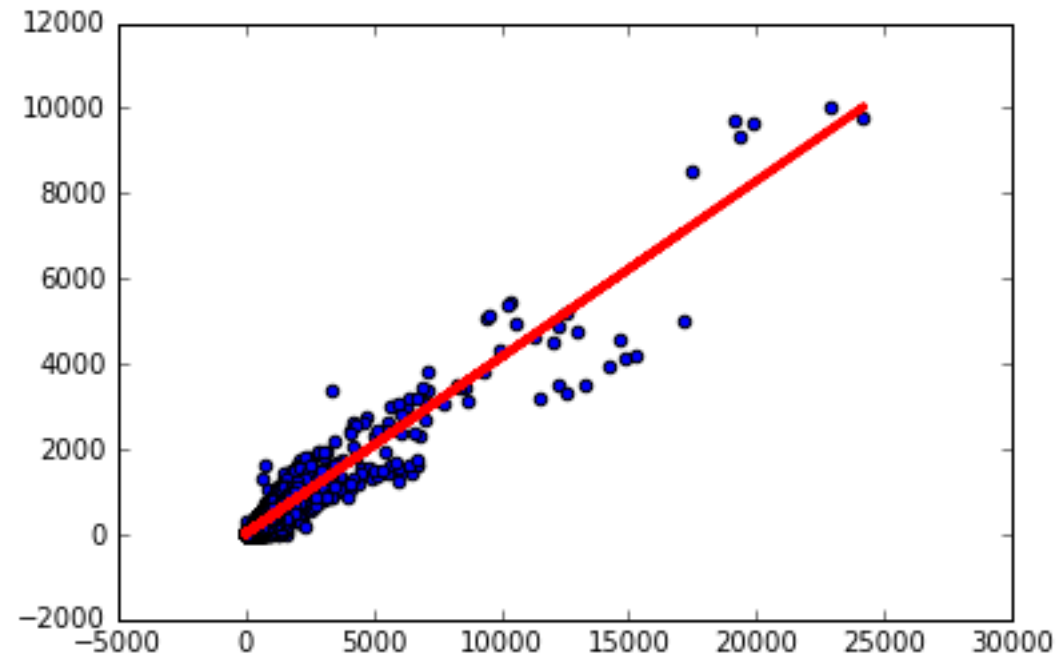
## Example:

```
from sklearn import linear_model
#select a column for x data and put it in the right shape
xdat = my_data["NCD_0"].reshape((20020,1))
ydat = np.transpose(my_data["AI_0"]) #change into column vector
reg = linear_model.LinearRegression()
reg.fit (xdat, ydat)

plt.scatter(xdat,ydat)
plt.plot(xdat, reg.predict(xdat), color='red', linewidth=2)
print reg.predict(100)
```

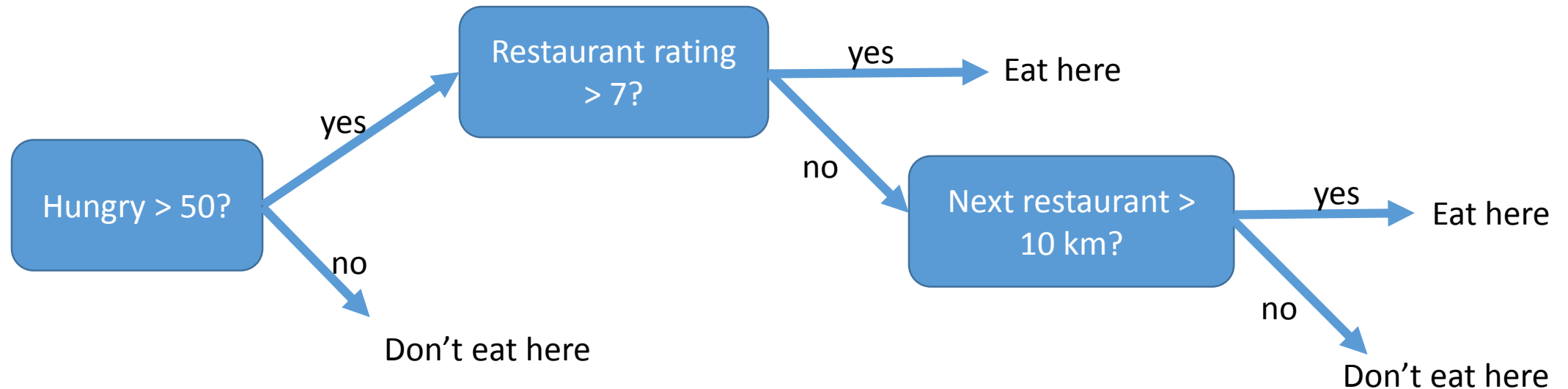
# Linear Regression (3)

Example:



# Decision Tree

- Decide which class an instance belongs to by learning simple decision rules
- At each step, decide which variable best splits the data instances



# Decision tree (2)

## Example:

```
from sklearn import tree
#Plot histogram of data to see distribution of the data
plt.hist(my_data["NAD_6"], bins=100)
plt.show()
#Make a variable representing the correct classes to be predicted
class_NAD6 = np.array(my_data["NAD_6"]>100).reshape((20020,1))
x2dat = my_data["AT_0"].reshape((20020,1))
```

# Decision tree (2)

## Example:

```
#Make instance of classifier and fit data
dt = tree.DecisionTreeClassifier()
dt.fit(x2dat,class_NAD6)
dt.classes_ #Which classes have been identified?
dt.feature_importances_ #How important is each feature?
dt.score(x2dat, class_NAD6)
#Try the same using NCD_0 as a feature instead of AT_0
dt.fit(xdat,class_NAD6)
dt.score(xdat,class_NAD6)
```

# Decision tree (3)

## Example:

```
#Split into a train and a testset
x2dat_train = x2dat[:10010]
x2dat_test = x2dat[10010:]
class_train = class_NAD6[:10010]
class_test = class_NAD6[10010:]
#Fit using training set, score using testset. Compare scores
dt.fit(x2dat_train,class_train)
dt.score(x2dat_test,class_test)
dt.score(x2dat_train,class_train)
```

# Machine learning continued

- Try some of the following algorithms (popular and well-documented on scikit-learn page):
  - Support vector machines (SVM)
  - Nearest Neighbors
  - Random forest
- Example datasets are also provided in the sklearn package:

```
from sklearn import datasets
```



# The art of debugging

- Syntax errors: there's something wrong with the syntax of your code. The system will point you to the place where the error was detected.

```
print "apple
```

- Exceptions: error trying to execute the code because your instructions are impossible or prohibited. The exception will usually give you helpful information why it ran into an error.

```
print fruitdish (when fruitdish has not been defined)
```

```
1000/0
```

- Bugs: code runs but doesn't actually do what you intended

# The art of debugging (2)

- Check for each step whether the variable(s) match what you expect, especially before and after calling a function
- Debugger
- Try using the “logging” module  
(<https://docs.python.org/2/howto/logging.html>)

```
import logging
import numpy as np
b = 2
answer = np.power(b, 3)
logging.debug('%d to the power of 3 is %d' % (b, answer))
```

# The art of debugging (3)

- Search online if you're not sure what your message means
- Ask/view questions on StackOverflow
- Check the documentation (if you're using a module)
- Open-source software: check open issues
- Write functions that throw exceptions if they are called in an inappropriate way



**KEEP  
CALM  
AND  
ASK  
QUESTIONS**

To keep going:

<https://www.codingame.com/>

<https://www.codeeval.com/>

<https://www.codecademy.com/>

[yz.fang1@gmail.com](mailto:yz.fang1@gmail.com)

“if you can do it for joy, you can do it for ever”

– Stephen King

How do you create tomorrow?