Algorithmic Approaches for Biological Data, Lecture #1

Katherine St. John

City University of New York
American Museum of Natural History

20 January 2016

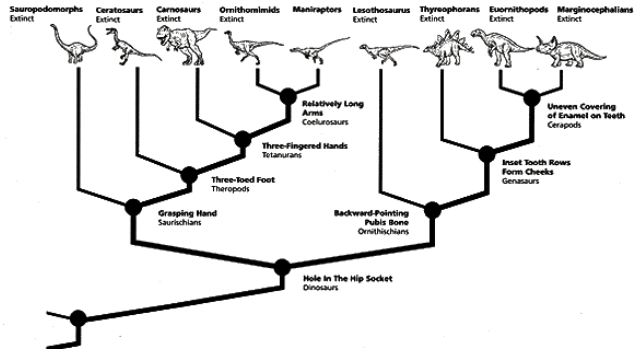
Outline

- Course Overview
- Introduction to Python Programs:
  - printing and simple functions
  - definite loops
  - Problem solving and the design process
- Recap

Katherine St. John

- Professor, City University of New York
- Research Associate, American Museum of Natural History
- Associate Academic Director, BridgeUp:STEM
- Postdoctoral Work at University of Texas and University of Pennsylvania
- PhD, UCLA
- MA, Johns Hopkins University
- AB, Smith College

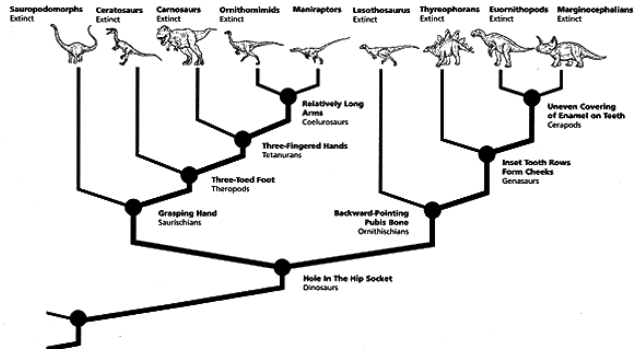
Katherine St. John



- Professor, City University of New York
- Research Associate, American Museum of Natural History
- Associate Academic Director, BridgeUp:STEM
- Postdoctoral Work at University of Texas and University of Pennsylvania
- PhD, UCLA
- MA, Johns Hopkins University
- AB, Smith College

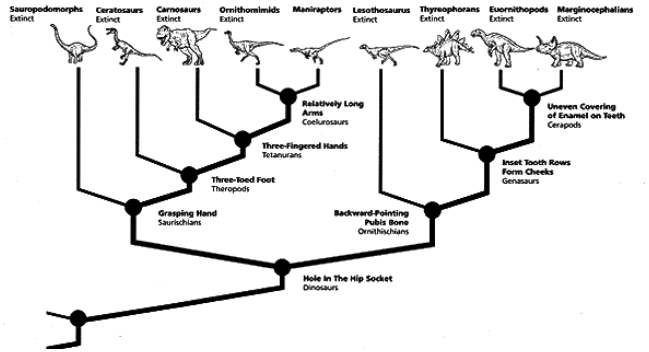
Katherine St. John

- Professor, City University of New York
- Research Associate, American Museum of Natural History
- Associate Academic Director, BridgeUp:STEM
- Postdoctoral Work at University of Texas and University of Pennsylvania
- PhD, UCLA
- MA, Johns Hopkins University
- AB, Smith College



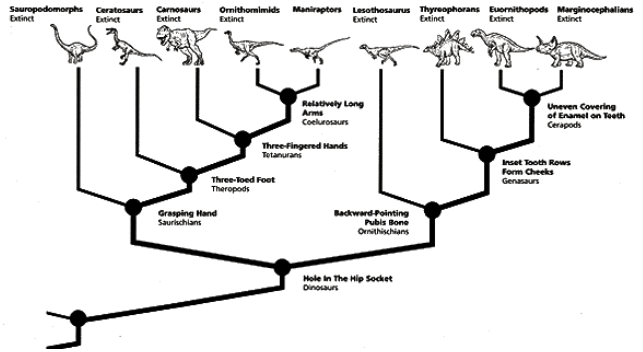
Katherine St. John

- Professor, City University of New York
- Research Associate, American Museum of Natural History
- Associate Academic Director, BridgeUp:STEM
- Postdoctoral Work at University of Texas and University of Pennsylvania
- PhD, UCLA
- MA, Johns Hopkins University
- AB, Smith College

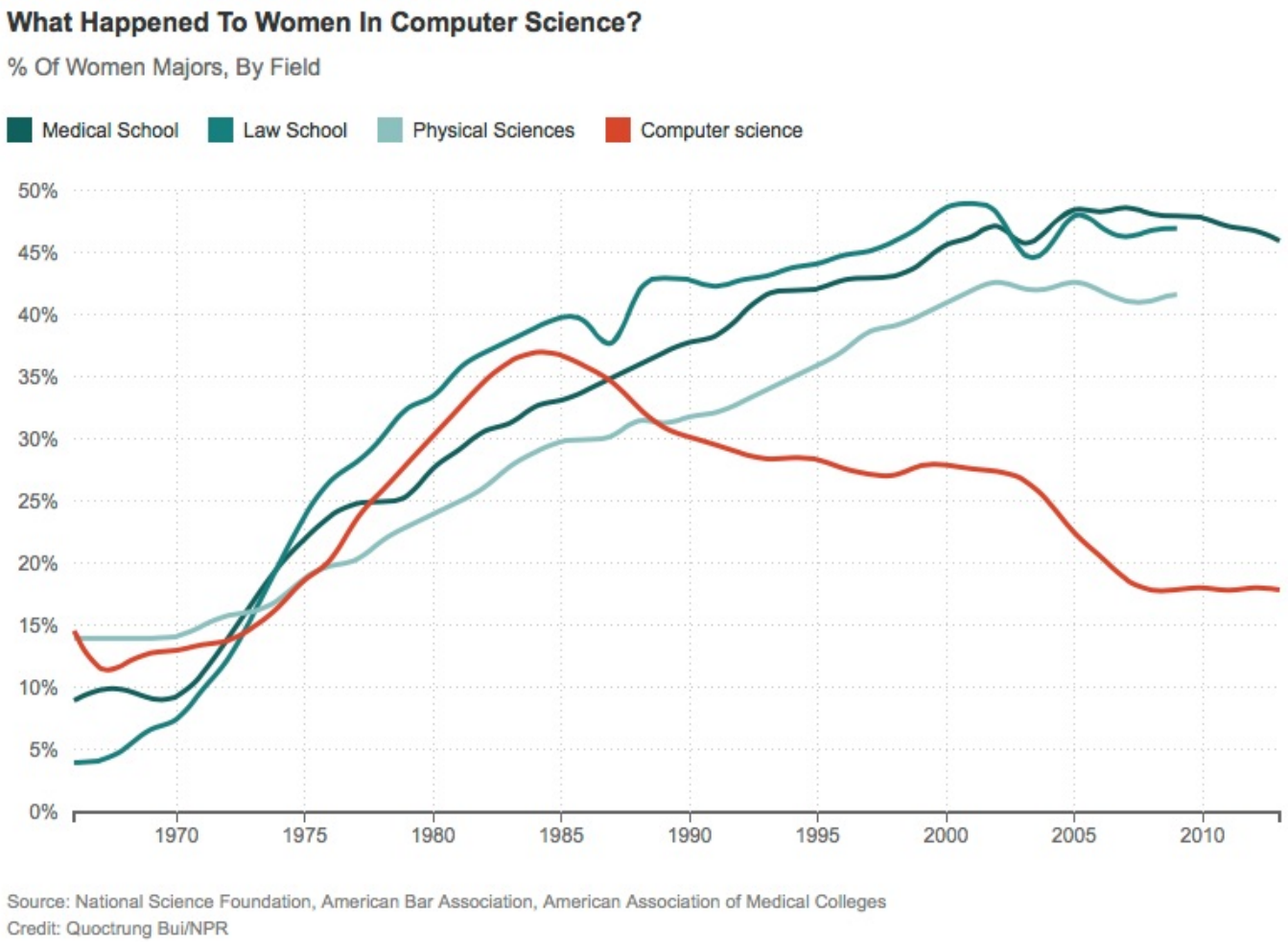


Katherine St. John

- Professor, City University of New York
- Research Associate, American Museum of Natural History
- Associate Academic Director, BridgeUp:STEM
- Postdoctoral Work at University of Texas and University of Pennsylvania
- PhD, UCLA
- MA, Johns Hopkins University
- AB, Smith College

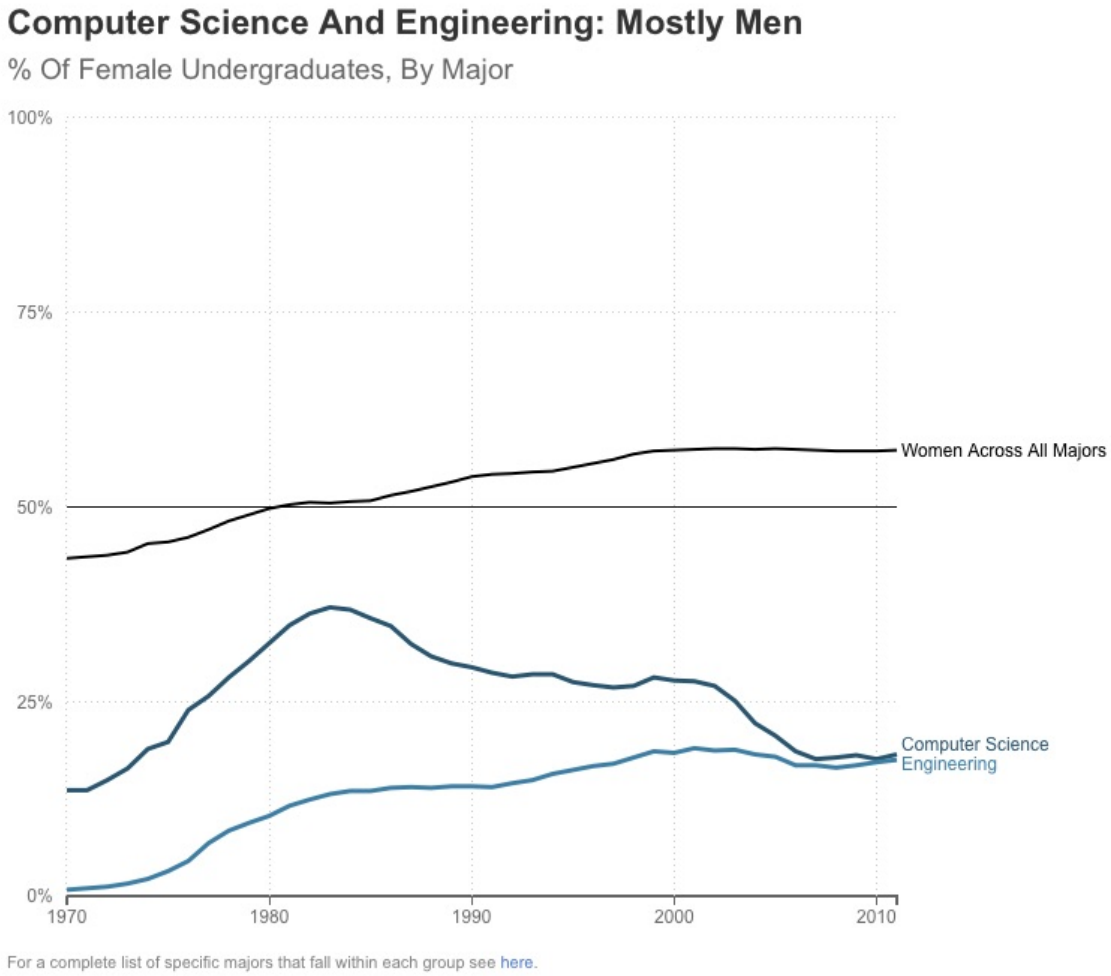


Historical Trends: Women Majors



NPR Planet Money, October 2014

Historical Trends: Computer Science & Engineering



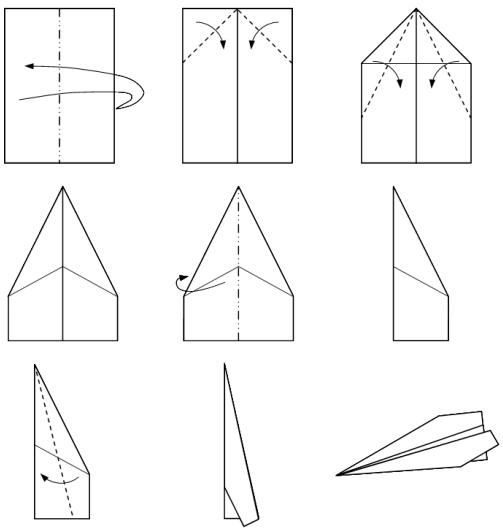
NPR Planet Money, October 2014

Course Website and Syllabus

All course material available at:

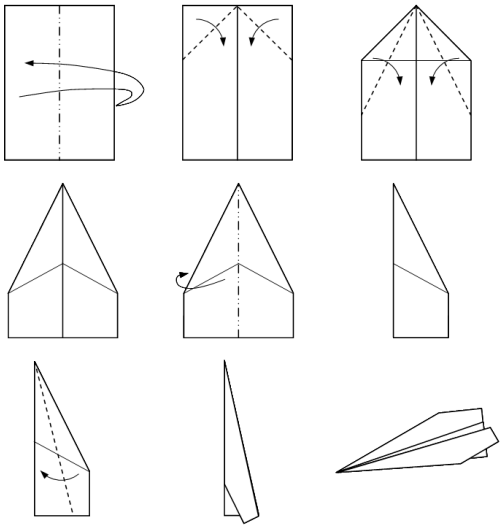
`comet.lehman.cuny.edu/stjohn`

Algorithms



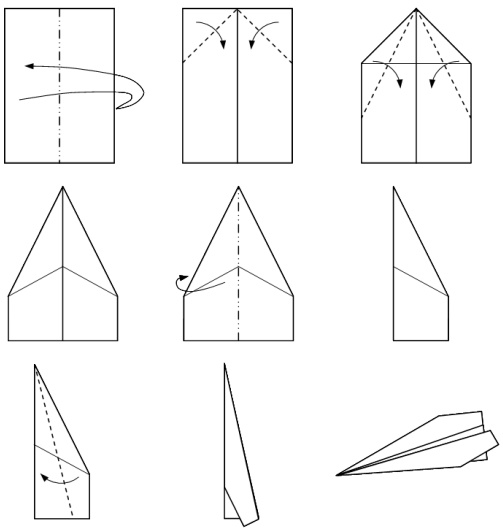
- An algorithm is a step-by-step directions to perform some task.

Algorithms



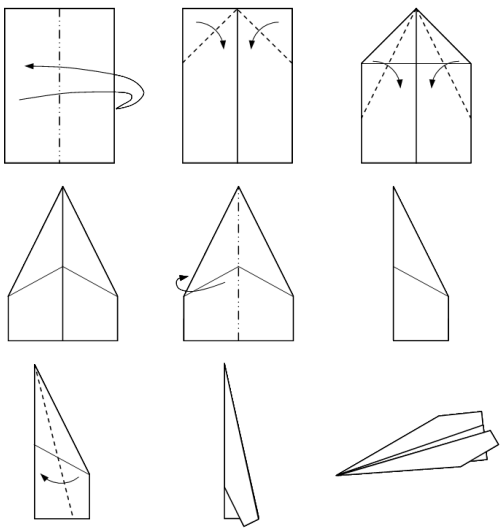
- An algorithm is a step-by-step directions to perform some task.
- Precision of language matters.

Algorithms



- An algorithm is a step-by-step directions to perform some task.
- Precision of language matters.
- Finding and fixing unexpected behavior is called debugging)

Algorithms



- An algorithm is a step-by-step directions to perform some task.
- Precision of language matters.
- Finding and fixing unexpected behavior is called debugging)
- Altering directions to improve the solutions is a key technique (called step-wise refinement)

Using Python

- Python 2.7: Part of the default installation on your laptops

Using Python

- Python 2.7: Part of the default installation on your laptops
- To launch, go to: *Application* > *Utilities* and click on `Terminal`. A window will pop up, type: `idle`)

Using Python

- Python 2.7: Part of the default installation on your laptops
- To launch, go to: *Application* > *Utilities* and click on `Terminal`. A window will pop up, type: `idle`)
- `IDLE` is a simple integrated development environment included with all Python distributions.

First Python Program

```
>>> print "Hello world"
```

First Python Program

```
>>> print "Hello world"
```

- ">>>": python prompt
- "print": prints to the terminal

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

Second Python Program

- "import turtle": includes turtle drawing

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws
- "for": repeats a statement

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws
- "for": repeats a statement
- "tess.right(x)": turns right x degrees

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws
- "for": repeats a statement
- "tess.right(x)": turns right x degrees
- "tess.forward(x)": moves forward x steps

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws
- "for": repeats a statement
- "tess.right(x)": turns right x degrees
- "tess.forward(x)": moves forward x steps
- "tess.clear()": clears the screen

Second Python Program

```
>>> import turtle
>>> tess = turtle.Turtle()
>>> for i in range(4):
    tess.forward(100)
    tess.right(90)
```

- "import turtle": includes turtle drawing
- "tess": an object that draws
- "for": repeats a statement
- "tess.right(x)": turns right x degrees
- "tess.forward(x)": moves forward x steps
- "tess.clear()": clears the screen
- "tess.home()": return to center

Saving Files

Retyping into the Python shell gets tedious, making refining code difficult:

- To create a new file, choose *New File* from the *File* menu.

Saving Files

Retyping into the Python shell gets tedious, making refining code difficult:

- To create a new file, choose *New File* from the *File* menu.
- Type your Python commands into the file window.

Saving Files

Retyping into the Python shell gets tedious, making refining code difficult:

- To create a new file, choose *New File* from the *File* menu.
- Type your Python commands into the file window.
- Save as a `.py` file (e.g. `first.py`) to allow color-coded text.

Saving Files

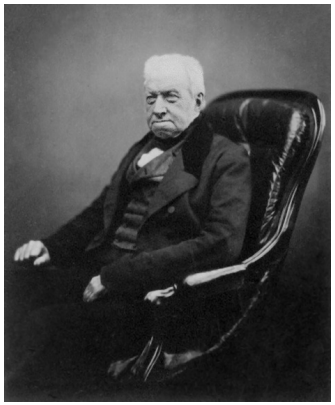
Retyping into the Python shell gets tedious, making refining code difficult:

- To create a new file, choose *New File* from the *File* menu.
- Type your Python commands into the file window.
- Save as a `.py` file (e.g. `first.py`) to allow color-coded text.
- Run your program by choosing *Run Module* from the *Run* module.

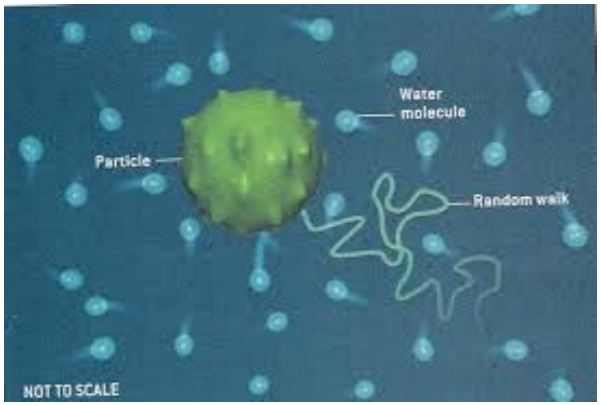
Challenges

- Draw a hexagon.
- Draw an octagon.
- Draw a 5-pointed star.
- Print your name to the screen 100 times.

Modeling Brownian Motion



Robert Brown



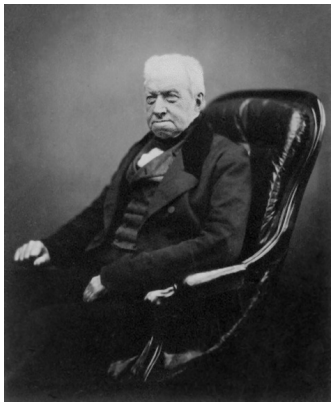
Brownian Motion

Albert Einstein

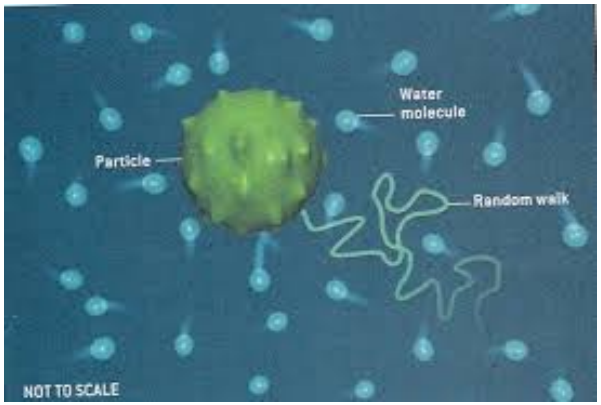
(Images from wikipedia and *Scientific American*)

- 1827: Robert Brown observed pollen grains moving under a microsope

Modeling Brownian Motion



Robert Brown



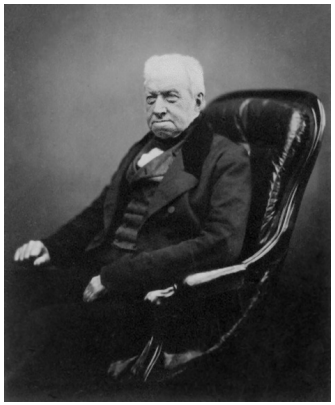
Brownian Motion

Albert Einstein

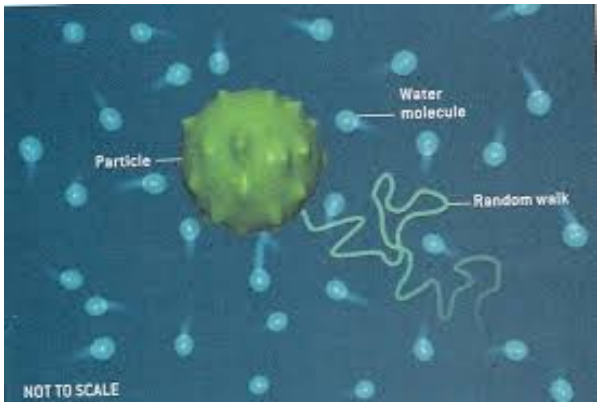
(Images from wikipedia and *Scientific American*)

- 1827: Robert Brown observed pollen grains moving under a microsope
- 1905: Albert Einstein explains the pollen's movement is due to the movement of individual water molecules.

Modeling Brownian Motion



Robert Brown



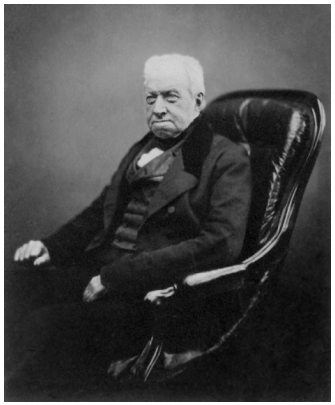
Brownian Motion

Albert Einstein

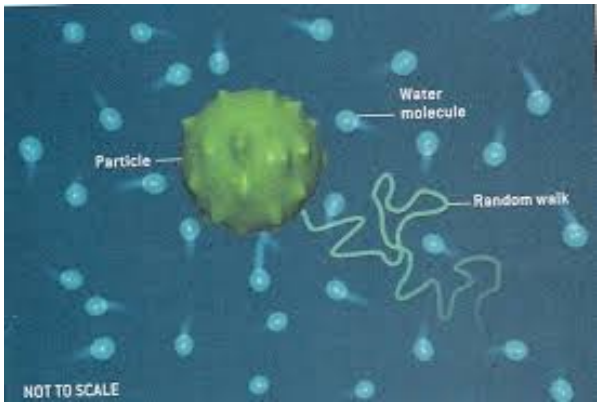
(Images from wikipedia and *Scientific American*)

- 1827: Robert Brown observed pollen grains moving under a microsope
- 1905: Albert Einstein explains the pollen's movement is due to the movement of individual water molecules.
- Brownian motion refers to the random motion of particles in water, as well as the underlying mathematical model.

Modeling Brownian Motion



Robert Brown



Brownian Motion

Albert Einstein

(Images from wikipedia and *Scientific American*)

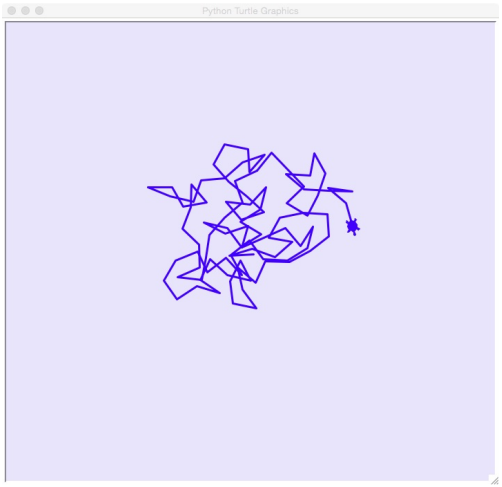
- 1827: Robert Brown observed pollen grains moving under a microsope
- 1905: Albert Einstein explains the pollen's movement is due to the movement of individual water molecules.
- Brownian motion refers to the random motion of particles in water, as well as the underlying mathematical model.
- Good model for the scattering of light and diffusion of substances

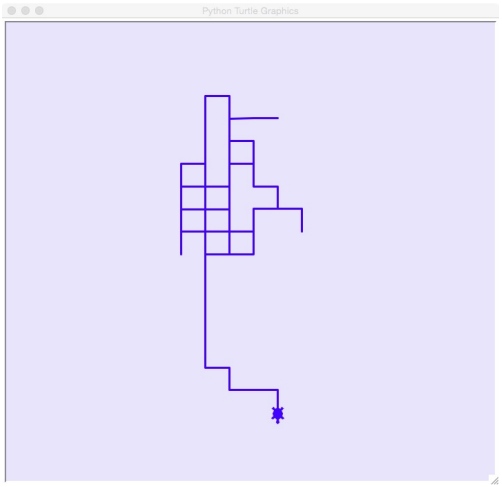
Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

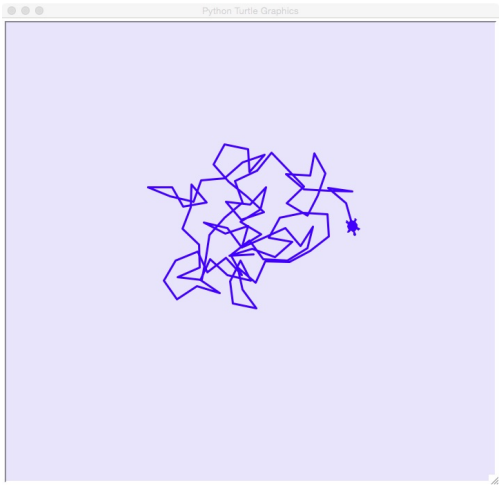
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

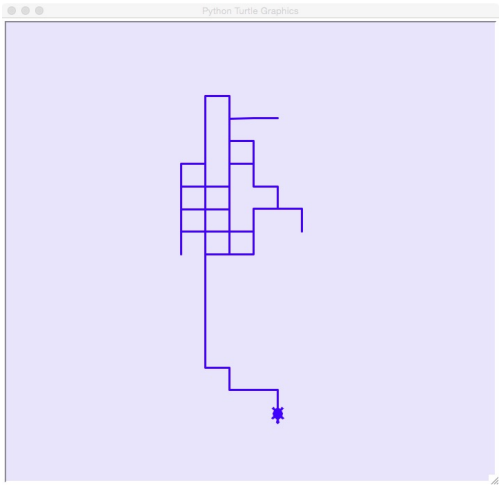
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

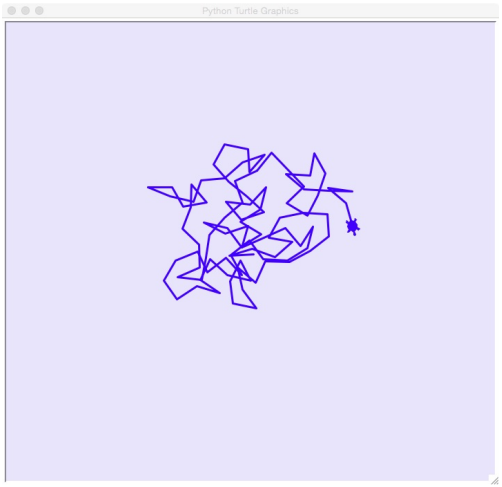
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

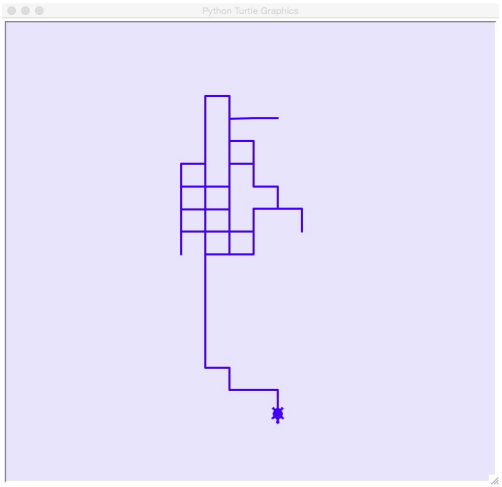
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

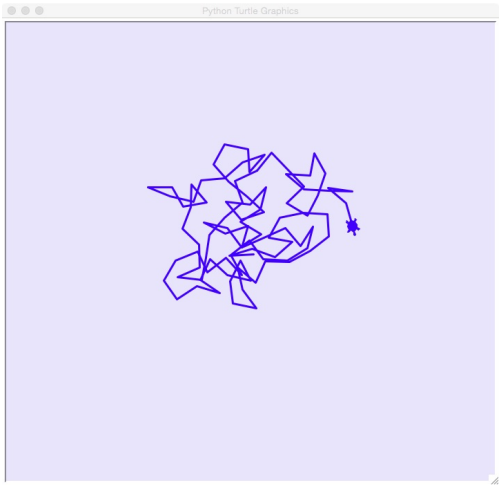
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

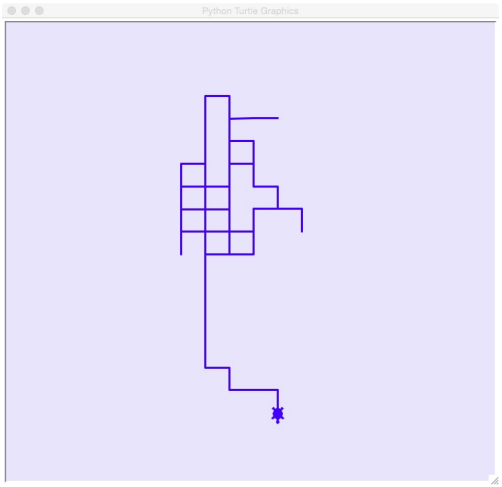
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?
  - Hypothesize what will happen?

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

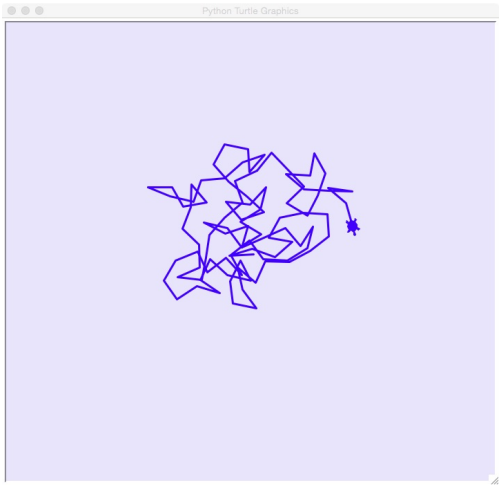
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

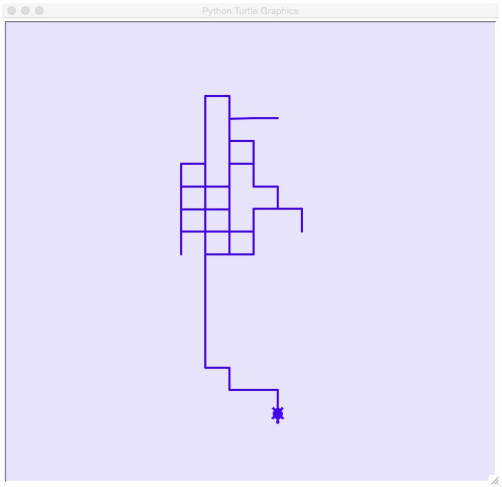
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?
  - Hypothesize what will happen?
  - Do 10 simulation runs.

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

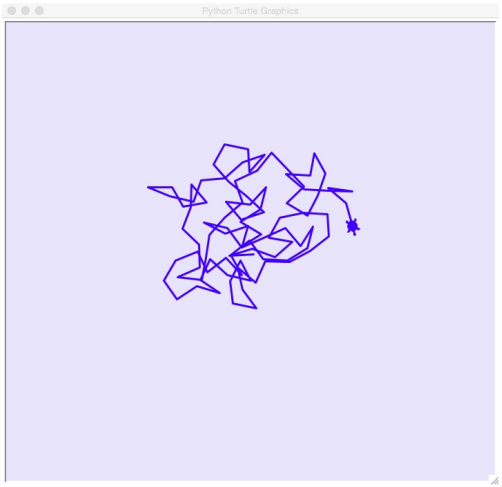
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

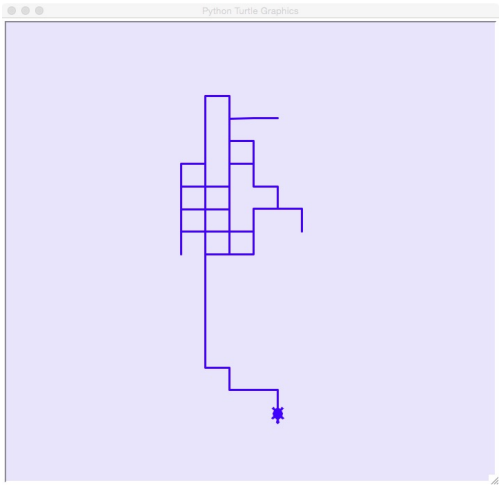
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?
  - Hypothesize what will happen?
  - Do 10 simulation runs.
  - What is the farthest? Nearest? Average?

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





- With a partner, fill in the program:

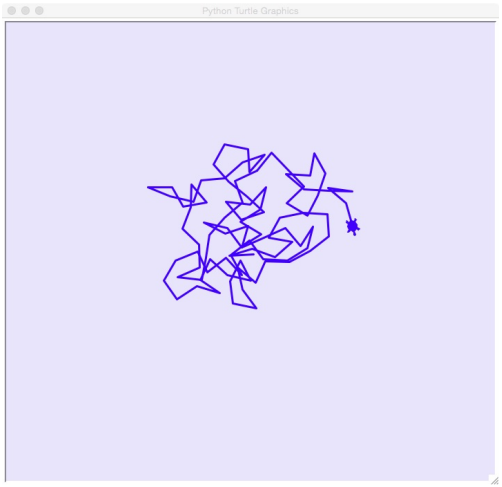
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

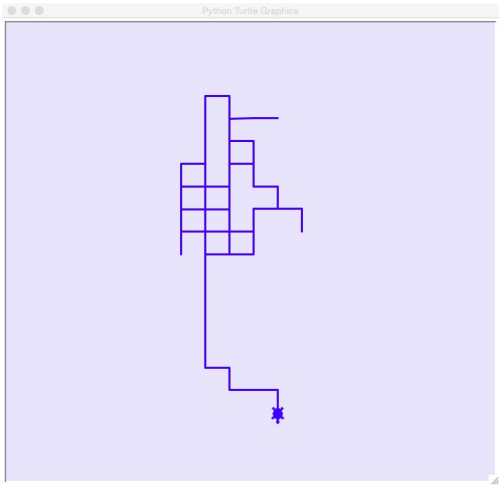
1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?
  - Hypothesize what will happen?
  - Do 10 simulation runs.
  - What is the farthest? Nearest? Average?
- Grid Random Walks: Change the direction line to: `right(random.randomrange(0,360,90)`

Modeling Brownian Motion

Random walks are used to simulate Brownian Motion.





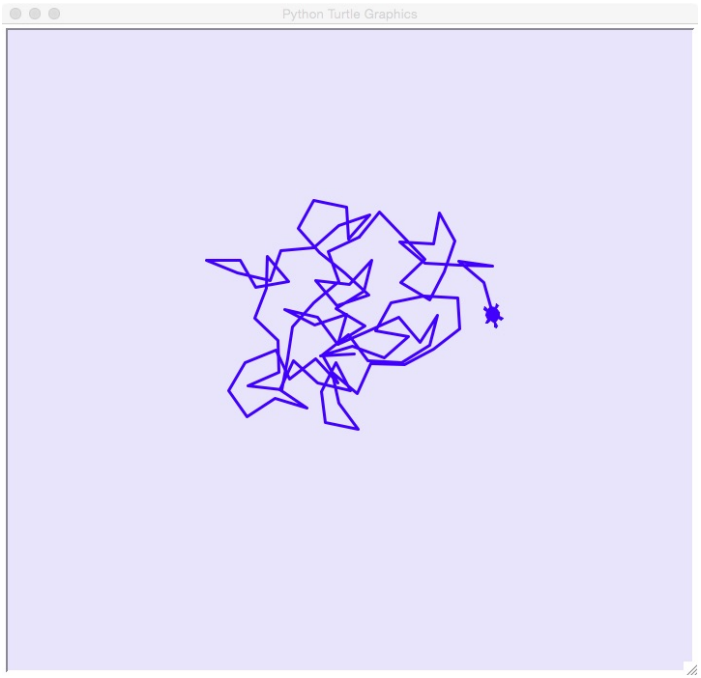
- With a partner, fill in the program:

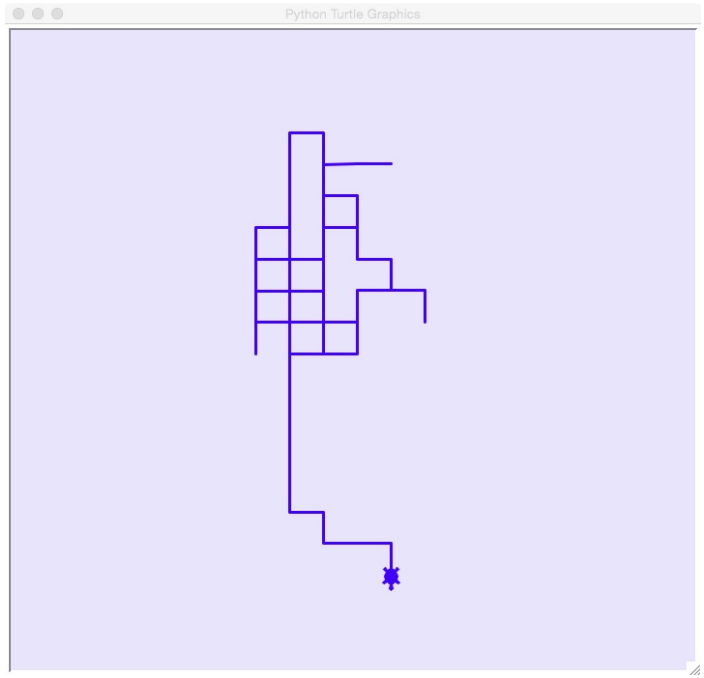
```
import random #Useful functions for random
import turtle #Turtle drawing functions
```

1. Create a turtle
2. For 100 steps
3. Move forward 10
4. Turn in a random direction: `right(random.randrange(0,360))`
5. Print out final x and y of turtle: `print tess.pos()`

- Implement and test your design.
- Experiment: How far does your turtle travel?
  - Hypothesize what will happen?
  - Do 10 simulation runs.
  - What is the farthest? Nearest? Average?
- Grid Random Walks: Change the direction line to: `right(random.randomrange(0,360,90)`
- 1D Random Walks: Change the direction line to: `right(random.randomrange(0,360,180)`

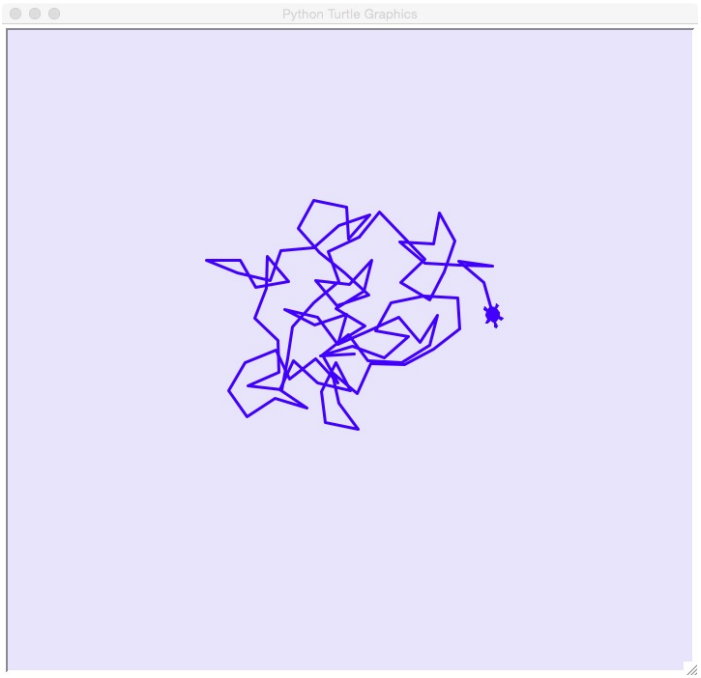
Recap

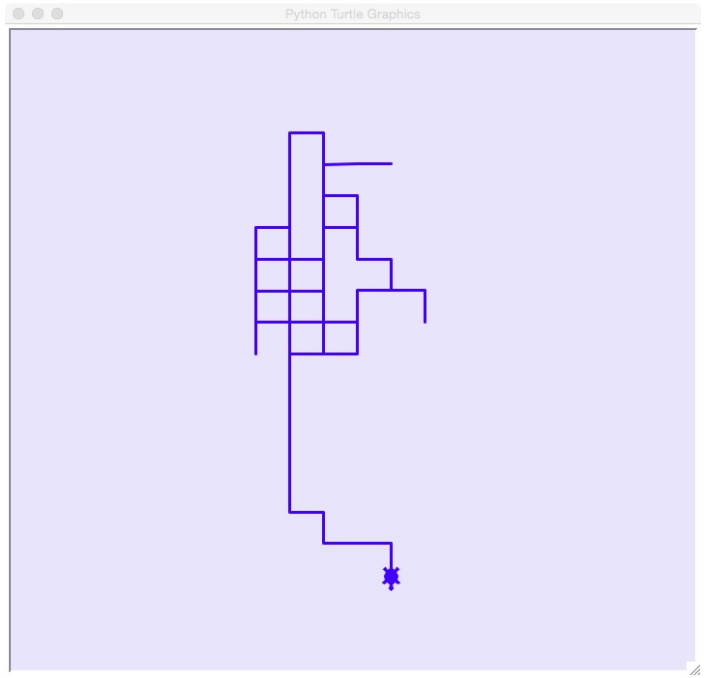




- Lab session at 3pm today (bring laptop!).

Recap





- Lab session at 3pm today (bring laptop!).
- Challenges available at `rosalind.info` (use emailed link to access course page).