CS101: Intro to Computing
Spring 2016

Lecture 20
Administrivia

- Homework 10 due Friday
- Midterm 2 April 18th at 7pm
REVIEW
```python
x=[4,2,6]
x=x.sort()
try:
    print(len(x))
except:
    print(type(x))
```

a) TypeError
b) 3
c) list
d) NoneType
```python
x=np.zeros((3,3))
for i in range(3):
    for j in range(3):
        x[i][j]=i*j+j
```

```
A
0  0  0
1  2  3
2  4  6

B
0  0  0
0  2  4
0  4  8

C
0  1  2
0  2  4
0  3  6
```
x=np.zeros((3,3))
for i in range(3):
    x[i][i]=1
    for j in range(3):
        if i<j:
            x[i][j]=2

A

1 2 2
0 1 2
0 0 1

B

1 0 0
2 1 0
2 2 1

C

2 2 1
2 1 0
1 0 0
```python
x = np.zeros((3, 3))
for i in range(3):
    x[i][0] = i
    for j in range(1, 3):
        x[i][j] = x[i][j-1] + 1
```
HOMEWORK HELP
State variables

- A collection of variables describing the current state of the system.
- Describe all information we are interested in.
- Describe all information needed to determine the future state.
Model pseudocode

set constants
state=initial_state
while not simulation_finished:
    state=update(state)
import numpy as np

# Parameters of simulation
n=100  # number of data points to plot
start=0.0  # start time of simulation
end=1.0  # ending time of simulation
g=-9.8  # acceleration of gravity

# State variable initialization
t=np.linspace(start,end,n+1)  # time in seconds
y=np.zeros(n+1)  # height in meters
v=np.zeros(n+1)  # velocity in m/s (v0=0m/s)
y[0]=1.0  # initialize height to 1m

for i in range(1,n+1):
    v[i]=v[i-1]+g*(t[i]-t[i-1])
    y[i]=y[i-1]+v[i-1]*(t[i]-t[i-1])
    if y[i] <= 0:  # glass has hit the ground
        v[i]=0
        y[i]=0
Time step size

• How much time passes between iterations of our simulation.

• What is $\Delta t$?

• Determines the resolution/fidelity of our model.

• Smaller step size means more accurate reproduction of reality.

• Smaller step size means longer run time.
v=0.0
y=1.0
g=-9.8
t=0
dt=???

while(y>0.0):
    t+=dt
    y+=v*dt
    v+=g*dt

Which of these values will produce the fastest simulation?

a)  1
b)  .1
c)  .01
d)  .001
Multiple Simulations

• In HW10, we are running many simulations

• Create a 2D array for state variables
  – Each row tracks a different simulation (angle)
  – Each column tracks one step in simulation
# Parameters of simulation

n=1000          # number of data points to plot
m=20            # number of balls to drop
start=0.0       # start time of simulation
end=2.0         # ending time of simulation
g=-9.8          # acceleration of gravity

# State variable initialization

t=np.linspace(start,end,n+1)    # time in seconds
y=np.zeros((m,n+1),dtype=np.float64)            # height in meters
v=np.zeros((m,n+1),dtype=np.float64)            # velocity in m/s

for i in range(m):
    y[i][0]=i+1

for i in range(m): # glass number
    for j in range(1,n+1): # time number
        if y[i][j-1]>0:
            y[i][j]=y[i][j-1]+v[i][j-1]*(t[j]-t[j-1])
            v[i][j]=v[i][j-1]+g*(t[j]-t[j-1])
        else:
            y[i][j]=0
            v[i][j]=0
Course Summary (so far…)

1. Python fundamentals
2. Data wrangling
3. Data visualization
4. Simulation
5. Random processes
6. Optimization
PSEUDORANDOM NUMBERS
Pseudorandom numbers

- PRNG - a program that generates seemingly random numbers
- Starting from an initial input (seed) PRNG generates “unpredictable” numbers
- Next input is generated from previous input
- numpy implements “Mersenne Twister” PRNG
Rand

- Creates an array sampled from uniform distribution on interval [0,1)
- Arguments are dimensions of array

```python
x = np.random.rand(10000)
plt.hist(x)
plt.show()
```
Rand

• Creates an array sampled from uniform distribution on interval $[0,1)$
Matshow

- Pyplot function to display a 2d array as an image
Randn

- Creates an array sampled from **standard normal distribution**
  - Mean 0 and variance 1
- Arguments are dimensions of array

```python
x=np.random.randn(10000)
plt.hist(x)
plt.show()
```
Randn

- Creates an array sampled from standard normal distribution
Other distributions

- Lognormal
- Poisson
- Binomial
- Rayleigh
- Gamma
- Geometric
- Chi-square
- Dirichlet
- Laplace
- Beta
- Gamma
- Gumbel
- Hypergeometric
- Cauchy
- Weibull
- And more!
Choice

- Generates a random sample from a 1-D array

```python
x = np.arange(1, 7)
c = np.random.choice(x)
```
Choice

• Can generate samples without replacement

```python
x = np.arange(1, 53)
c = np.random.choice(x, size=5, replace=False)
```
Shuffle

- Randomly reposition elements of a 1-D array
- Creates a random permutation of the array elements

```python
x = np.arange(1, 53)
np.random.shuffle(x)
```
Example
Descriptive Statistics

- Can compute descriptive statistics like:
  - Min, max, percentile
  - Mean, median, standard deviation

```python
x = np.random.rand(1000)
print(np.median(x))
print(np.mean(x))
```
Example

- A random walk in 2 dimensions
- Start in middle
- Take one step in a random direction: north, south, east, or west