Some random asides...

```
import random
from random import *
# allows use of dir(random) and help(random)
# all random functions are now available!

def choice(L):
    # choose 1 element from the sequence L

def choice('mudd'):
    # ...or 1 character from a string

def choice(['cmc','scripps','pitzer','pomona'])

print(list(range(1, 5)))

>>> uniform(41.9, 42.1)
42.08010107642389

How would you get a random integer from 0 to 99 inclusive?
```

A random function...

```
from random import *
def guess(hidden):
    # Tries to guess our "hidden" number
    compguess = choice(list(range(100)))

    if compguess == hidden:  # at last!
        print('I got it!')
    else:
        guess(hidden)
```

Monte Carlo in action

```
def countDoubles(N):
    # Argument: the number of dice rolls to make
    # Result: the number of doubles seen"
    if N == 0:
        return 0  # zero rolls, zero doubles...
    else:
        d1 = choice([1, 2, 3, 4, 5, 6])
        d2 = choice(list(range(1, 7)))

        if d1 == d2:
            return 1 + countDoubles(N - 1)  # COUNT IT!
        else:
            return 0 + countDoubles(N - 1)  # don't count it

N is the total number of rolls

```

How many doubles will you get in N rolls of 2 dice?
Monte Carlo Monty Hall

```python
def MCMH(init, sors, N):
    if N == 0: return 0
    else:
        forward(100)  # one side
        left(360 / N)  # turn 360/N
        poly(n-1, N)  # draw rest
        return 1 + MCMH(init, sors, N - 1)

if init == przDoor and sors == 'stay': result = 'Spam!'
elif init == przDoor and sors == 'switch': result = 'pmfp.'
else:
    result = 'pmfp.'
print 'You get the', result
```

Python's Etch-a-Sketch

```python
import time
from turtle import *

def draw():
    # define it!
    shape('turtle')
    # pause
time.sleep(2)
    # drawing...
    width(5)
    left(90)
    forward(50)  # pixels!
    right(90)  # degrees!
    backward(50)
    down() or up()  # is the pen on the "paper"?
    color('darkgreen')
    tracer(1) or tracer(0)

# run it!
reset()
draw()
```

An example closer to home

```
Start

Class (W)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
</tr>
<tr>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
</tr>
<tr>
<td>49</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

An overworked 5C student (S) leaves H/S after their “late-night” breakfast—or lunch. Each moment, they randomly stumble toward class (W) or the dorm (E)

Once the student arrives at the dorm or classroom, the trip is complete.

Write a program to model and analyze! this scenario...

Single-path recursion

```
def tri():  # define it!
    """A triangle!"""
    forward(100)
    left(120)
    forward(100)
    left(120)
    forward(100)
    left(120)

    # run now
    tri()
```

(1) Let’s tri this with recursion:

```
def tri(n):
    """A triangle!"
    if n == 0: return
    else:
        forward(100)  # one side
        left(120)  # turn 360/3
        tri(n-1)  # draw rest
```

(2) How about any regular N-gon?

```
def poly(n, N):
    """A polygon!"
    if n == 0: return
    else:
        forward(100)  # one side
        left(360 / N)  # turn 360/N
        poly(n-1, N)  # draw rest
```
def chai(dist):
    """Mystery!""
    if dist < 5:
        return
    forward(dist)
    left(90)
    forward(dist / 2)
    right(90)
    right(90)
    forward(dist)
    left(90)
    left(90)
    forward(dist / 2)
    right(90)
    backward(dist)

lab ~ hw2pr1

fractal art

def chai(50):

svtree(trunkLength, levels)

The Koch curve

snowflake(100, 0)  snowflake(100, 1)  snowflake(100, 2)
snowflake(100, 3)  snowflake(100, 4)  snowflake(100, 5)
**Quiz**

Name(s): __________________  A few *random* thoughts...

```python
from random import *
choice([1, 2, 3, 2])
```

What are the chances this returns a 2?

```python
choice(list(range(1, 5)) + [4, 2, 4, 2])
```

What are the chances of this returning a 4?

```python
choice('1,2,3,4')
choice(['1,2,3,4'])
choice('[1,2,3,4]')
```

What's the most likely return value here?

```python
choice(list(range(5)))
```

Is this more likely to be even or odd (or same)?

```python
uniform(-20.5, 0.5)
```

What're the chances of this being >= 0?

```python
choice(0, 1, 2, 3, 4)
choice(['list(range(5))'])
choice[list(range(5))]
```

Which **two** of these 3 are *syntax errors*?

Also, what does the **third** one—the one syntactically correct—actually *do*?