Iterators and Generators Homework - Solution

Problem 1

Create a generator that generates the squares of numbers up to some number N.

In [10]:
```
def gensquares(N):
    for i in range(N):
        yield i ** 2
```

In [11]:
```
for x in gensquares(10):
    print x
```

```
0
1
4
9
16
25
36
49
64
81
```

Problem 2

Create a generator that yields "n" random numbers between a low and high number (that are inputs). Note: Use the random library. For example:

In [6]:
```
import random

random.randint(1,10)
```

Out[6]: 6

In [7]:
```
def rand_num(low,high,n):

    for i in range(n):
        yield random.randint(low, high)
```

In [9]:
```
for num in rand_num(1,10,12):
    print num
```

```
6
7
6
9
9
4
3
7
8
1
6
1
```

Problem 3

Use the iter() function to convert the string below

In [17]:
```
s = 'hello'

s = iter(s)

print next(s)
```

```
h
```

Problem 4

Explain a use case for a generator using a yield statement where you would not want to use a normal function with a return statement.

If the output has the potential of taking up a large amount of memory and you only intend to iterate through it, you would want to use a generator. (Multiple answers are acceptable here!)

Extra Credit!

Can you explain what bonus is in the code below? (Note: We never covered this in lecture!)

In [18]:
```
my_list = [1,2,3,4,5]

gencomp = (item for item in my_list if item > 3)

for item in gencomp:
    print item
```

```
4
5
```

Hint google: generator comprehension is!

Great Job!