Samples Programming with Python SOLUTIONS

1. The full Python inputs and results are shown below. Please note that there may be some slight rounding errors present, and that the results may not be formatted exactly as you would find in Python.

Here is the program:

```
from __future__ import division
from pylab import *

def petersRule(r):
    # Implementing the rule for Peter's Conjecture
    if r % 2 == 0 :
        return r/2
    else :
        return (r+3)*2

# The parameters of our experiment
init = 10
maxCount = 27

# Initialising the parameters of our experiment
z = init
count = 0

# Creating a table of values
print "Sequence of values when evaluating Peter's Conjecture"
print "-----------------------------------------------------"
print z

while (z != 1) and (count < maxCount) :
    z = petersRule(z)
    print z
    count = count+1

# Print a blank line
print

# Determining whether we've reached the expected value of 1
if z == 1 :
    print "It took", count, "iterations to reach 1 when starting with a value of", init
else :
    print "Could not reach 1 within", maxCount, "iterations when starting with a value of", init
```

Here is the output from running the program:

```
Sequence of values when evaluating Peter's Conjecture
-----------------------------------------------------
10
5.0
16.0
8.0
4.0
2.0
1.0
```

```
It took 6 iterations to reach 1 when starting with a value of 10
```

2. The full Python inputs and results are shown below. Please note that there may be some slight rounding errors present, and that the results may not be formatted exactly as you would find in Python.

Here is the program:

```
from __future__ import division
from pylab import *

def petersRule(p):
    # Implementing the rule for Peter's Conjecture
    if p % 4 == 0 :
        return p/4
    else :
        return 4*p*3

# The parameters of our experiment
init = 9
maxCount = 25

# Initialising the parameters of our experiment
l = init
count = 0

# Creating a table of values
print "Sequence of values when evaluating Peter's Conjecture"
print "-----------------------------------------------------"
print l

while (l != 1) and (count < maxCount) :
    l = petersRule(l)
    print l
    count = count+1

# Print a blank line
print

# Determining whether we've reached the expected value of 1
if l == 1 :
    print "It took", count, "iterations to reach 1 when starting with a value of", init
else :
    print "Could not reach 1 within", maxCount, "iterations when starting with a value of", init
```

Here is the output from running the program:

```
Sequence of values when evaluating Peter's Conjecture
-----------------------------------------------------
9
108
27.0
324.0
81.0
972.0
243.0
2916.0
```

```
729.0
8748.0
2187.0
26244.0
6561.0
78732.0
19683.0
236196.0
59049.0
708588.0
177147.0
2125764.0
531441.0
6377292.0
1594323.0
1.913188e+07
4782969.0
5.739563e+07

Could not reach 1 within 25 iterations when starting with a value of 9
```

3. The full Python inputs and results are shown below. Please note that there may be some slight rounding errors present, and that the results may not be formatted exactly as you would find in Python.

Here is the program:

```
from __future__ import division
from pylab import *

def petersRule(n):
    # Implementing the rule for Peter's Conjecture
    if n % 2 == 0 :
        return n/2
    else :
        return (3+n)*2

# The parameters of our experiment
init = 5
maxCount = 24

# Initialising the parameters of our experiment
e = init
count = 0

# Creating a table of values
print "Sequence of values when evaluating Peter's Conjecture"
print "----------------------------------------------------"
print e

while (e != 1) and (count < maxCount) :
    e = petersRule(e)
    print e
    count = count+1

# Print a blank line
```

```
print

# Determining whether we've reached the expected value of 1
if e == 1 :
    print "It took", count, "iterations to reach 1 when starting with a value of", init
else :
    print "Could not reach 1 within", maxCount, "iterations when starting with a value of", init
```

Here is the output from running the program:

```
Sequence of values when evaluating Peter's Conjecture
-----------------------------------------------------
5
16
8.0
4.0
2.0
1.0

It took 5 iterations to reach 1 when starting with a value of 5
```