Gregory Jerian
8/21/14
Period 4

Ch 1 Programming Exercises

1. a. Hello, world!
b. Hello world!
c. 3
d. 3.0
e. 5
f. 5.0
g. 23
h. 2 + 3 = 5
i. 6
j. 8
k. 0.6666666666666666

2.
```
>>> def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1-x)
        print(x)
```

3.
```
>>> def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 2.0 * x * (1-x)
        print(x)
```

The first chaos program begins to spit out more and more random-seeming outputs over time, while the second chaos program slowly moves towards 0.5. While the first program gets more “chaotic” over time, the second program does the opposite and becomes more orderly over time.

4.
```
>>> def main():
    print("This program illustrates a chaotic function")
```

```
x = eval(input("Enter a number between 0 and 1: "))
for i in range(20):
        x = 2.0 * x * (1-x)
        print(x)
```

5.
```
>>> def main():
print("This program illustrates a chaotic function")
x = eval(input("Enter a number between 0 and 1: "))
n = eval(input(“How many numbers should I print? “))
for i in range(n):
        x = 2.0 * x * (1-x)
        print(x)
```

6.
```
>>> def main():
print("This program illustrates a chaotic function")
x = eval(input("Enter a number between 0 and 1: "))
n = eval(input(“How many numbers should I print? “))
for i in range(n):
        x = 3.9 * x * (1-x)
        print(x)
```

```
>>> def main():
print("This program illustrates a chaotic function")
x = eval(input("Enter a number between 0 and 1: "))
n = eval(input(“How many numbers should I print? “))
for i in range(n):
        x = 3.9 * (x - x * x)
        print(x)
```

```
>>> def main():
print("This program illustrates a chaotic function")
x = eval(input("Enter a number between 0 and 1: "))
n = eval(input(“How many numbers should I print? “))
for i in range(n):
        x = 3.9 * x - 3.9 * x * x
        print(x)
```

All three of these programs create the same outputs.