**Notes based off of Charles Severance’s “Python for Informatics: Exploring Information**

**Print** a string or an integer:

* + - * print (’Hello World!’)
      * print(‘4’)

Set a **Value** to use later in the program:

* + - * x=4

x

4

* + - * y=8

y

8

To check the **type** of a value:

* + - * type(‘Hello World!’)

<class ‘str’>

* + - * type(4)

<class ‘int’>

* + - * type(4.3)

<class ‘float’>

**Keywords**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| and | del | from | not | while |
| as | elif | global | or | with |
| assert | else | if | pass | yield |
| break | except | import | print | max |
| class | exec | in | raise | min |
| continue | finally | is | return | len |
| def | for | lambda | try | range |

**Operators**:

* + - * Addition +
      * Subtraction -
      * Multiplication \*
      * Division / or % (to check remainder)
      * Exponation \*\*
      * Compares and is either true or false ==
      * See book page 31 & 40 for list of Comparison Operators

Print a **Remainder**:

* + - * remainder=12%5

print (remainder)

2

**Newline**:

\n

**Comments** start with #

**Conditional Statement**:

* + - * if x > 0 :

print ( ‘x is positive’ )

* + - * elif x<0:

print (‘x is negative’)

* + - * + *elif stands for else if…you can have an infinite chain of elifs*
      * else:

print (‘x is equal to zero’)

**Min** and **Max** values:

* + - * *these functions give us the largest and smallest values in a list (cap sensitive)*
      * max(‘Hello World!’)

‘r’

* + - * min(‘Hello World!’)

‘ ‘

Number of **Characters** in a string:

* + - * len(‘Hello world!’)

12

**Conversions**:

**Value to Integer**:

int(2.3)

2.3

int(3.9999)

3

**Integers to Floating-Point numbers**:

float(32)

32.0

**Argument to String**:

str(32)

’32’

**Generating Random Numbers**:

import random

for i in range (10):

x=random.random()

print(x)

*the number entered in parenthesis is the number of numbers generated*

random.randint(1,100)

35

t=[1,3,5,7,9]

random.choice(t)

1

See page 46 for **Math Functions** and uses for the math module

* + - * math.pi

3.141592653589793

**Void Functions** are functions that perform an action but don’t return a value. They do not store values assigned to them.

To **Count** the number of times a letter appear in a string:

* string=‘dani is a dork’

count=0

for letter in string:

if letter==‘d’:

count=count+1

print(count)

2

**In** operator:

* ‘f’ in ‘metropolis’

False

* ‘g’ in ‘gregarious’

True

* ‘fred’ in ‘gregarious’

False