Name: ______ SOLUTION ______________ **Section:** 1 2 3 4

Use this practice to help you prepare for the Paper-and-Pencil portion of Test 1. ***Answer all questions.*** Make additional notes as desired. ***Not sure of an answer?*** Ask your instructor to explain in class and revise as needed then.

Throughout, where you are asked to "circle your choice", you can underline or circle it (whichever you prefer).

1. Consider the ***secret*** function defined to the right. What are the values of:

```python
def secret(x):
    y = (x + 1) ** 2
    return y
```

   a. `secret(2)` _______ **9** __________

   b. `secret(secret(2))` _____ **100** _______

2. Consider the ***mystery*** function defined to the right. What are the values of:

```python
def mystery(x, y):
    result = x + (3 * y)
    return result
```

   a. `mystery(5, 2)` ____**11** _____

   b. `mystery(2, 5)` ____**17**______

3. Consider the code snippets defined below. They are contrived examples with poor style but will run. For each, what does it print when *main* runs? (Each is an independent problem.)

```python
def main():
    x = 5
    foo(x)
    print(x)

def foo(x):
    print(x)
    return x ** 3
```

```python
def main():
    x = 5
    y = foo(x)
    print(y)

def foo(x):
    x = 10
    print(x)
    return x ** 3
```

```python
def main():
    x = 5
    x = foo(x)
    print(x)

def foo(x):
    print(x)
    return x ** 3
```

Prints: _______ **5** ______ ___________**10**___ ____ **5** __________

______ **5**_______ ______ **1000** _ ____ **125** _________

4. What is the value of each of the following expressions?

`7 // 4`	`1`
`3.0 // 4.0`	`0`
`3 / 4`	`0.75`
`7 % 2`	`1`
`7 ** 2`	`49`
`'fun' + 'ny'`	`'funny'`
`'hot' * 5`	`'hothothothothot'`

5. For each of the following code snippets, what does it print? (Write each answer directly below its code snippet.)

```
for j in range(0, 8, 2):
    print(j)
```

```
0

2

4

6
```

```
a = 10
for k in range(3, 6):
    a = a + k
    print(a, k)
```

```
13  3

17  4

22  5
```

```
b = 0
for k in range(10, 2, -1):
    if (k % 3) == 2:
        b = b + 1
        print(b, k)
print(b)
```

```
1  8

2  5

2
```

6. For each of the following Boolean expressions, indicate whether it evaluates to ***True*** or ***False*** (circle your choice):

True [✓ *False*]	`not (5 < 7)`
True [✓ *False*]	`(7 < 5) or not (5 < 7)`
[✓ *True*] *False*	`(3 != 4) and (3 == 3)`
[✓ *True*] *False*	`(6 <= 6) or (3 == 2)`
True [✓ *False*]	`(6 <= 6) and (3 == 2)`
True [✓ *False*]	`not not False`

7. For each of the following, write a **range** expression that produces the given sequence:

```
4, 5, 6, 7, 8              range(4, 9)

40, 50, 60, 70, 80         range(40, 90, 10)
                              or  range(40, 81, 10)   [or similar]

-6, -5, -4                 range(-6, -3)

-6, -7, -8                 range(-6, -9, -1)
```

8. What gets printed when ***main*** is called in the program shown to the right? (Pay close attention to the order in which the statements are executed. ***Write the output in a column to the left of the program.***)

Output

```
2  3

2  3

2  3
```

```
def main():
    a = 2
    b = 3

    foo1()
    print(a, b)

    foo2(a, b)
    print(a, b)

    foo3(a, b)
    print(a, b)

def foo1():
    a = 88
    b = 99

def foo2(a, b):
    a = 400
    b = 500

def foo3(x, y):
    x = 44
    y = 55
```

9. True or False: As a ***user*** of a function (that is, as someone who will ***call*** the function), you *don't need to know how the function is* ***implemented****;* you just need to know the ***specification*** of the function. [✓ **True**] **False** (circle your choice)

10. List ***two*** reasons why functions are useful and important.

Reason 1: **They help organize the code, which makes it easier to get correct and maintain.**

Reason 2: **They allow for code re-use, by allowing the function to be called multiple times with different values for the parameters.**

11. ***float*** versus ***int***:

a. Write two Python constants – one an integer (**`int`**) and one a floating point number (**`float`**) – that clearly shows the difference between the **`int`** and **`float`** types.

82 **13.793** **[the *float* has a decimal point]**

b. A Python **`int`** can have an arbitrarily large number of digits. [✓ ***True***] ***False***
(circle your choice)

c. A Python **`float`** can represent an arbitrarily large number. ***True*** [✓ ***False***]
(circle your choice)

d. There is a limit to the number of significant digits a Python **`float`** can have. [✓ ***True***] ***False***
(circle your choice)

12. ***int*** versus ***str***: What does each of the following code snippets print or cause to happen if the user types **5** in each case? (Write each answer to the side of its code snippet.)

```
x = input('Enter an integer: ')
print(x * 3)
```

555

```
y = int(input('Enter an integer: '))
print(y * 3)
```

15

```
z = input('Enter an integer: ')
print(z / 3)
```

Raises an exception (error). That is, it causes the program to break at the attempt to do division on a string.

13. Consider a function whose name is ***print_string*** that takes two arguments as in this example:

```
print_string('Robots rule!', 4)
```

The function should print the given string the given number of times. So, the above function call should produce this output:

```
Robots rule!
Robots rule!
Robots rule!
Robots rule!
```

Write (in the space below) a complete implementation, ***including the header (def) line***, of the above ***print_string*** function.

```
def print_string(s, n):
    for k in range(n):
        print(s)
```