

Name: _____ **SOLUTION** _____

1. Consider the code snippet shown to the right.

- (a) What is the **type** of the object to which the name (i.e., variable) *greeting* refers? **str** (*string is OK too*)
- (b) What is the **type** of the object to which the name (i.e., variable) *x* refers? **int** (*integer is OK too*)
- (c) What is the **value** of *x* **after** the snippet runs? **5**
- (d) What is the **value** of *y* **after** the snippet runs? **2**
- (e) What is the **value** of *z* **after** the snippet runs? **60**

```
greeting = "hello"
x = 27 // 5
y = 27 % 5
z = 2 ** 3
z = z + 12
z = z * 3
```

2. Consider the code snippet shown to the right.

Underline the parameters and **circle** the arguments.

3. The following steps happen when calling a function.

Number them in the order they occur (1, 2, 3, 4).

```
def main():
    a = 10
    b = 5
    print(foo(a, b)) arguments (should be CIRCLED)

def foo(a, b): parameters (should be UNDERLINED)
    return (a ** 2) + (b ** 2)
```

2 Run the code in the function's definition.

1 Jump to the function's definition, assigning the function's parameters, if any, to the arguments in the function call.

4 Continue code execution from where the function was called.

3 Return the value to where the function was called.

4. Consider the program shown to the right. In the box beside it, show what that program prints (i.e., displays) when it runs.

```
def main():
    print("hi")
    foo()
    x = bar()
    print(3 * x)
    print("bye")

def foo():
    print("ok:", bar())
    print("foo")

def bar():
    print("bar")
    return 8

main()
bar()
```

Output:

```
hi
bar
ok: 8
foo
bar
24
bye
bar
```

5. Consider the program shown to the right. In the box below, show what that program prints (i.e., displays) when it runs.

Output:

112

24

```
def main():
    print(foo(10, 2))
    print(foo(2, 10))

def foo(a, b):
    return (a ** 2) + 10 + b

main()
```

6. Fill in the blanks below. Besides a name, every class has what three things?

A _____ **constructor method** _____ to create and initialize instances of the class,
 _____ **methods** _____ that do stuff, and
 _____ **instance variables** _____ that store data.

7. In this problem assume that there is a name **turtle** that has been defined to refer to a SimpleTurtle.

(a) Suppose that there is a **method** in the SimpleTurtle class called **forward** that takes the distance to move forward as a parameter. Write a statement that would make **turtle** move 100 pixels forward using the **forward** method.

turtle.forward(100)

(b) Now suppose that there is a **function** called **another_forward** that takes a SimpleTurtle and the distance to move forward as its TWO parameters. Write a statement that would make **turtle** move 100 pixels forward using the **another_forward** function.

another_forward(turtle, 100)

8. In the box to the right, define a function named **silly** that takes two parameters, both SimpleTurtle objects. The function makes the first SimpleTurtle move 100 pixels backward and makes the second SimpleTurtle turn left 60 degrees. The function returns the sum of the speeds of the two SimpleTurtle objects.

```
def silly(turtle1, turtle2):
    turtle1.backward(100)
    turtle2.left(60)
    return turtle1.speed + turtle2.speed
```