Name: ______________________________

Student #: __________________________

Introduction to Python Worksheet

Please use Jupyter to write Python code so that you can confirm your answers to these questions. We suggest that you make a prediction for what will happen before testing it in Jupyter, and then test in Jupyter to see if your understanding was correct.

1. What type of data is each of the following values?

Value	Type (`float`, `int`, `str` or `bool`)
`10.0`	
`'pi'`	
`17`	
`'3.14'`	
`True`	
`'False'`	
`-0.001`	

2. What are the first three things you like to look for to figure out what type of data a value is? (There's no one right answer, but it's good to think about your strategies.)

Please use Jupyter to write Python code so that you can confirm your answers to the rest of the programming questions. We suggest that you make a prediction for what will happen before testing it in Jupyter, and then test in Jupyter to see if your understanding was correct.

3. For each of the following expressions, indicate what value it will evaluate to.

Expression	Value
`3 * 4 + 7`	
`3 * 1.5`	
`4 * 1.0`	
`4.0 + '1'`	
`4.0/2.0 + 1.0`	
`10/2 + 3`	

4. Circle all of the expressions below that will evaluate to the value `True`.

```
2 > 3                       10.0 > 9.999           3 == 3.0

5.0 == 5.0                  1.0 + 1.0 == 2.0       not False

not (10 < 20)               4.0 == 3 + 1           3.0 != 4.0

False or True               False and True         True and True

(4 < 8 and 3 + 1 == 4) or 1 != 2                   2 > 1 or 2 > 4
```

5. The following code is evaluated in order. In the boxes on the right, provide the value that the expression on the left will evaluate to.

```
a = 3
a + 2        →  □
a = a + 1
b = 2
a == 4       →  □
a + b        →  □
a = a + b
b            →  □
a            →  □
```

6. The following code is evaluated in order. In the boxes on the right, provide the value that the expression on the left will evaluate to.

```
x = 1
y = 4       →
x + 4
y = 2
y == 4      →
x + y       →
x = y + 3
x           →
y           →
```

7. For each of the following expressions, indicate what value the expression will be evaluated to, or if it will produce an error.

Expression	Value
`'hello' + 'world'`	
`'hello' * 3`	
`'hello' + 3.0`	
`'hello' * '3'`	
`'world'[2]`	
`'world'[:2]`	
`'world'[2:]`	
`'helloworld'[-1]`	

8. Using `str1`, `str2` and `str3` and basic string operations, write an expression that will evaluate to each of the following strings.

```
str1 = 'University '
str2 = 'of '
str3 = 'British Columbia'
```

a. `'University of British Columbia'`

b. `'Universe'`

9. What is the value stored in result when each of the following code fragments have run?

a.

```
value = 13
result = 'default'

if value == 7:
    result = 'lucky'
```

```
elif value == 13:
    result = 'unlucky'

else:
    result = 'boring'
```

b.

```
temp = 35.0
result = 'default'


if temp < 10.0:
    result = 'too cold'
elif temp > 25.0:
    result = 'too hot'
elif temp > 30.0:
    result = 'way too hot'
else:
    result = 'just right'
```

10. Define a function that adds 2 to a number.

11. Using the Facts API from class, define a function that accepts a number and returns a fact about a number. The output of your function should also include information about the number that was used to generate the fact.

    For example, imagine the fact returned about the number 0 is the string "is the additive identity". To make the statement a bit easier to understand, we also want to include which number the fact is talking about. Our goal is to have our function return something like "The number 0 is the additive identity".

11. a. Write a line of code that calls the function you wrote in problem 11 and assigns the result to a new variable.

    b. Write a line of code that calls the function you wrote in problem 11 with different inputs. Assign the result to a new variable.

    c. Write a line of code that calls the function from problem 11 twice (with different inputs). Connect the two sentences with the word "and".

12. Define a function that takes a string parameter `s` and returns `'short'` if `s` is 3 characters long or less, `'medium'` if it is 7 characters or less, and `'long'` otherwise. (Check out the Language Module to look for a function that gives you the length of a string.)

13. a. Write a line of code to call the function you defined in problem 13 with some string value as its argument.

    b. Write a line of code to call the function you defined in problem 13 with a different string value that will produce a different returned result.

    c. Write a line of code to call the function you defined in problem 13 with a third string value that will produce a third, different returned result.

14. Define a function that takes a string parameter `word` and returns its plural according to the following rules:
    - if the string ends in ‘s’, make no changes
    - if the string ends in ‘o’, add ‘es’
    - if the string ends in ‘f’, remove the f, and add ‘ves’
    - otherwise, add ‘s’

    Also include enough calls to your function to check that it is working correctly in all the possible cases. (In Jupyter, you’ll likely want to put each of these calls in its own cell so that running the cells shows you all the results of these calls.)

15. Write the step-by-step evaluation of the following expression. Use the notation from the Evaluation Rules document from the Language module on Canvas (underlining each expression that is being evaluated to a value).

```
3.0 * (10.0 / 5.0) + 10.0
```

16. Write the step-by-step evaluation of the line `foo(9.0/2.0)`:

```
def foo(x):
    if x > 5:
        return x*2
    elif x < 2:
        return x + 7
    else:
        return x

foo(9.0/2.0)
```

Notes:

- To trace evaluation of a function call: evaluate all of its argument expressions to values (like 20 rather than 4*5); then, go off to the side, make a note of what the parameters' values are, and trace the evaluation of the body of the function separately. When you get to a `return` and have traced evaluation of its expression to a simple value (like `return True` or `return 6`), go back to your original evaluation trace and replace the underlined function call with the returned value.

- To trace evaluation of an `if/elif/else` statement, find the branch (the `if`, `elif`, or `else`) to execute, if any, and trace through its body (the statements indented within that branch). In more detail: first trace evaluation of the `if`'s condition expression to a single value `True` or `False`. If `True`, replace the whole statement with body of the `if`, deleting any `elif` or `else`. If it's `False`, delete the `if` part of the statement and its body. If that removes the `if` statement entirely, you're done with that statement. If not, it will leave you with an `elif` or an `else`. If it's an `elif`, just change it into an `if` and keep going. If it's an `else`, replace the whole statement with the body of the `else`.

17. Is there anything you'd like to share with us so that we can help you achieve your goals in this class? For example, family or other commitments, personal needs such as needing to sit close to the front of class, etc.

18. What are you finding most helpful as you begin to learn how to program in Python?

19. What are you finding most difficult about learning Python?

---- If you are finished you can begin working on the pre-class work for next class ----