Introduction to Python Warmup Worksheet

Instructions:

1. Log into Grok.
2. Go to https://groklearning.com/learn/python-turtle-playground/2/0/
3. Read the directions for each task very carefully before you begin working on it.
4. Every time the directions say something like "take a screenshot", do it and then copy it into the proper space in this document.

This worksheet is intended to be both a review of all of the topics that we have covered so far in our Introduction to Python unit. Work through each task, being sure to type/run each bit of code yourself.

Problem 0: Identify Yourself

1. What are the names of the people in your group?

   >

2. Share this document with **all people** in the group and make them editors.

   ☐ Check the box when you're done

Problem 1: Which Path

Use the code below to answer the following questions and complete the required tasks.

```
x = 6
y = 10
word = ""
```

```
if y >= 10:
  word = word + "h"
else:
  word = word + "m"

if x + y == 16:
  word = word + "e"

if x == 7:
  word = word + "h"
elif y != 5:
  word = word + "y"
else:
  word = word + "f"

print(word)
```

BEFORE running the program, answer the following questions:

1. What is the data type (e.g., string, integer, or float) of the `x` variable?

   >

2. What is the data type of the `y` variable?

   >

3. What is the data type of the `word` variable?

   >

4. What is the difference between `=` and `==`?

   >

5. What do you think will be printed by the program?

   >

NEXT, copy the code into Grok and run it. Results will show up under the **output tab**.

1. Were your predictions correct? If not, what mistakes did you make?

   >

2. What values of `x` and `y` would result in the program printing `"meh"`?

   >

Problem 2: Going Up

Use the code below to answer the following questions and complete the required tasks.

```
x = 0
for i in range(6):
  x = x + i
print(x)
```

BEFORE running the program, answer the following questions:

1. This code is an example of the *accumulator pattern*.

   a. What is the *accumulator variable*?

      >

   b. Is there a *selection condition*? If so, which line is it?

      >

c. Which line is the *accumulator step*?

>

d. Which line is the *accumulation result*?

>

2. What is the data type (e.g., string, integer, or float) of the `x` variable?

>

3. What is the data type of the `i` variable?

>

4. Which of the two variables in the program is the *accumulator variable*? Why do you think this?

>

5. In the iteration table below, show what the value of `i` and `x` will be each time through the loop.

Iteration Count	**Value of `i`**	**Value of `x`**
before start of iteration	`N/A`	`0`
1	`0`	
2		
3		
4	`3`	
5		`10`
6		

6. What do you think will be printed by the program?

   >

NEXT, copy the code into Grok and run it. Results will show up under the **output tab**.

1. Were your predictions correct? If not, what mistakes did you make?

   >

2. What is a better name for the variable `x` in the program? **Make the change** in all the right places in your program.

   >

3. Modify the program so it says `The "accumulation result" is: X`. Note that there are double quotes in the string.

4. Copy **your updated code in Grok** and paste it here.

   ```

   ```

Problem 3: Grab Bag

Create a program that accomplishes each of the following tasks. Whenever you complete a program, copy **your code in Grok** and paste it under the proper section.

1. Write a program that asks the user for their name and then says `Hi NAME!`. Note that there's a `!` **at the end.**

   ```

   ```

2. In the previous problem, you had to use an “f-string”. Why was this necessary?

   >

3. Write a program that asks the user for a height and a width of a rectangle and says the area of the rectangle. Then, if the area is smaller than 10 the program says `That's a small rectangle!`. For example, if the user inputs height `2`, and width `3` the program should output: `6: That's a small rectangle!`. If the user inputs `5` and `11` the program should just output `55`.

   Note: no matter what, the area of the rectangle should be printed.

   ```

   ```

4. A very common mistake when writing a program that asks the user for a number is forgetting to “convert the input to an integer”. For example, a lot of people forget to do it in both the previous and the following problem. How do you convert a string to an integer and what happens if you forget to do this?

   >

5. Write a program that asks the user to enter a speed in miles per hour. Then, depending on what they say outputs `"30 is a perfect speed"`, `"less than 30 seems a bit slow"`, or `"more than 30 seems a bit fast"`.

   ```

   ```

6. In the previous problem, you likely had to use an if/elif/else. How could you have rewritten the program to just use if/else?

   >

7. Write a program that prints out all of the numbers from 0 to 100 (inclusive) and then prints `And that's how you count to 100 in CS`.

   ```

   ```

8. Write a program that asks the user to enter a sentence and counts how many `"e"`s there are in the sentence.

   Hint: You will need to use an iteration loop and an *accumulator variable*

   ```

   ```

Problem 4 (10pt bonus): On the Plus Side

Use the code below to answer the following questions and complete the required tasks.

```
coded_word = "L-k+a-m-q+b-r"
prev = "+"
x = ""
for c in coded_word:
  if prev == "+":
    x = x + c
  prev = c

print(x)
```

BEFORE running the program, answer the following questions:

1. This code is an example of the *accumulator pattern*.

   a. What is the *accumulator variable*?

>

b. Is there a *selection condition*? If so, which line is it?

>

c. Which line is the *accumulator step*?

>

d. Which line is the *accumulation result*?

>

2. What is the data type (e.g., string, integer, or float) of the `coded_word` variable?

>

3. What is the data type of the `x` variable?

>

4. What is the data type of the `c` variable?

>

5. What does the fourth line of the program do? Use the word “iterate” in your answer.

>

6. Which of the four variables in the program is the *accumulator variable*? Why do you think this?

>

7. In the iteration table below, show what the value of `c`, `prev`, and `x` will be through **just the first five iterations** of the loop.

Iteration Count	Value of `c`	Value of `prev`	Value of `x`
before start of iteration	`N/A`	`+`	`""`
1	`L`	`+`	
2			
3			
4			
5			

8. What word do you think will be output by the program?

   >

NEXT, copy the code into Grok and run it. Results will show up under the **output tab**.

1. Were your predictions correct? If not, what mistakes did you make?

   >

2. What is a better name for the variable `x` in the program? **Make the change** in all the right places in your program.

   >

3. Modify the program so it outputs the *accumulation result* in **all uppercase letters**.

4. Copy **your updated code in Grok** and paste it here.

   ```

   ```

Problem 5 (10pt bonus): Sentence Stats

Write a program that asks the user to enter a sentence and then prints out a series of statistics about that sentence including the number of uppercase letters, lowercase letters, vowels, and spaces. For example, given the input `"Hey there everyone!"`, the output would be:

```
Number of uppercase letters: 1
Number of lowercase letters: 15
Number of vowels: 7
Number of consonants: 9
```

Once you have completed the program, copy **your code in Grok** and paste it in the space below.

Hint: You will need to use the accumulator pattern with *multiple* accumulator variables
Hint: You will need to use the `isupper()` function.

```

```