Module 3 Graded Assessment

LATEST SUBMISSION GRADE

100%

Fill in the blanks of this code to print out the numbers 1 through 7. 1 / 1 point 1 number = 12 while number <= 7: Run print(number, end=" ") 3 Reset number = number+1 1 2 3 4 5 6 7 Correct Nice job! You're really getting the hang of what goes into the while loops! The show letters function should print out each letter of a word on a separate line. Fill in 1 / 1 point the blanks to make that happen. 1 def show_letters(word): 2 for x in word: 3 print(x) 4 Run show_letters("Hello") 5 Reset # Should print one line per letter Н

✓ Correct

Great job! You're working the "for" loops the way they're supposed to be done!

3. Complete the function digits(n) that returns how many digits the number has. For example: 25 has 2 digits and 144 has 3 digits. Tip: you can figure out the digits of a number by dividing it by 10 once per digit until there are no digits left.

1 / 1 point

```
def digits(n):
    1
    2
             count = 0
    3
             if n == 0:
    4
               return 1
    5
             while (n > 0):
                 count += 1
    6
    7
                 n = n//10
    8
             return count
    9
        print(digits(25))
                             # Should print 2
   10
        print(digits(144)) # Should print 3
   11
                                                                                   Run
   12
        print(digits(1000)) # Should print 4
                                                                                     Reset
   13
        print(digits(0))
                             # Should print 1
2
3
4
1
```

✓ Correct

Woohoo! You've cracked the code of writing code!

4. This function prints out a multiplication table (where each number is the result of multiplying the first number of its row by the number at the top of its column). Fill in the blanks so that calling multiplication table(1, 3) will print out:

1 / 1 point

123

246

369

```
1
        def multiplication_table(start, stop):
    2
            for x in range(start,stop+1):
    3
                 for y in range(start,stop+1):
    4
                     print(str(x*y), end=" ")
    5
                print()
                                                                                  Run
    6
    7
        multiplication_table(1, 3)
                                                                                     Reset
    8
        # Should print the multiplication table shown above
1 2 3
2 4 6
3 6 9
```

✓ Correct

Awesome! You've stepped up to the challenge of one of the more complex coding practices, nested loops!

5. The counter function counts down from start to stop when start is bigger than stop, and counts up from start to stop otherwise. Fill in the blanks to make this work correctly.

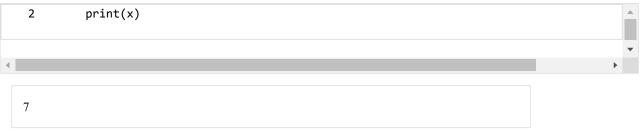
```
def counter(start, stop):
    2
            x = start
    3
            if x > stop:
                 return_string = "Counting down: "
    4
    5
                 while x >= stop:
    6
                     return string += str(x)
    7
                     if x>stop:
                         return_string += ","
    8
    9
                     x=x-1
   10
            else:
                 return_string = "Counting up: "
   11
   12
                 while x <= stop:
                     return_string += str(x)
   13
   14
                     if x<stop:</pre>
  15
                         return_string += ","
   16
                     x=x+1
   17
            return return_string
   18
        print(counter(1, 10)) # Should be "Counting up: 1,2,3,4,5,6,7,8,9,10" Run
   19
        print(counter(2, 1)) # Should be "Counting down: 2,1"
   20
                                                                                     Reset
        print(counter(5, 5)) # Should be "Counting up: 5"
Counting up: 1,2,3,4,5,6,7,8,9,10
Counting down: 2,1
Counting up: 5
```

Correct

You nailed it! You've figured out all of the situations that need to be considered!

6. The even_numbers function returns a space-separated string of all positive numbers that are divisible by 2, up to and including the maximum that's passed into the function. For example, even numbers(6) returns "2 4 6". Fill in the blank to make this work.

```
1  def even_numbers(maximum):
2    return_string = ""
3    for x in range(2,maximum+1,2):
4        return_string += str(x) + " "
5    return return_string.strip()
6
7  print(even_numbers(6)) # Should be 2 4 6
```

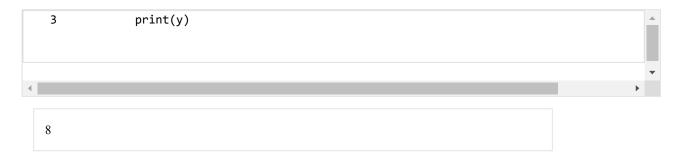


✓ Correct

You got it! The upper limit of a range isn't included, which means that the loop stops before reaching it. The increment is 3, so the loop stops when x reaches 7.

9. What is the value of y at the end of the following code?





✓ Correct

Great job! The upper limit of a range isn't included, which means that the outer loop goes up to 9, so the highest upper limit for the inner loop is 9, which is also not included.

 $10\cdot$ How does this function need to be called to print yes, no, and maybe as possible options to vote for?

1 / 1 point

```
1  def votes(params):
2   for vote in params:
3     print("Possible option:" + vote)
4
```

- votes("yes", "no", "maybe")
- votes(yes, no, maybe)
- votes([yes, no, maybe])
- votes(['yes', 'no', 'maybe'])

Correct

Excellent! This function is looking for one argument, and the list of strings is just one argument.